重视大脑的学习指南

Head First Mobile Web

（中文版）



一次构建，
到处运行



提供更大支持
（为用户）



利用地理定
位找到你的
位置



利用响应式Web设计
让网站变形



在小屏幕上让
页面减减肥



Lyza Danger Gardner & Jason Grigsby 著
林琪 刘晓兵 等译

O'REILLY® 中国电力出版社

Head First Mobile Web（中文版）

Mobile

从这本书能学到什么？

移动Web的使用在呈爆炸式增长。很快，人们会更愿意在手机和平板电脑而不是PC上浏览网页。你的企业需要一种移动策略，不过从哪里开始呢？《Head First Mobile Web（中文版）》会告诉你如何使用你熟悉的Web技术建立网站和应用，可以在各种任意大小的设备上工作。把你的HTML、JavaScript和CSS技术派上用场，然后优化你的网站，让它在要求很高的移动市场中有最出色的表现。在这个过程中，你会发现如何针对特定的设备来调整你的业务策略。

- 在越来越复杂的移动领域里自由驰骋。
- 同时采用技术和策略方法完成移动Web设计。
- 使用最新的开发技术，包括响应式Web设计，以及利用WURFL完成服务器端设备检测。
- 通过图片、谜题、故事和问答轻松学习。



为什么这本书如此与众不同？

我们认为，你的时间如此宝贵，不应过多地浪费在与新概念的斗争中。通过使用认知科学和学习理论的最新研究成果，你将享受一种多感官学习体验，本书采用了一种专门为你的大脑而设计的丰富格式娓娓道来，而不是长篇累牍地说教，让你昏昏欲睡。

“《Head First Mobile Web（中文版）》成功地展示了大量实用技术，所有读者都能立即上手，另外还提供了丰富的基础资源，使有经验的开发人员如虎添翼。Lyza和Jason针对当今这个移动Web开发的纷繁世界，以实效为宗旨为我们娓娓道来，还让我们有机会了解到该如何面向未来。”

——**Stephen Hay**，
Zero Interface
的创始人和负责人

“《Head First Mobile Web（中文版）》对这个复杂但令人兴奋的主题提供了完美的诠释。从一开始就着手实践，对于在Web设计领域探索时可能遇到的机会和挑战做了绝妙的介绍。”

——**Stephanie**和
Bryan Reiger，
Yiibu设计人员

ISBN 978-7-5123-4112-8

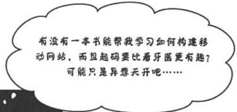
定价：68.00元

O'REILLY®
oreilly.com.cn
headfirstlabs.com

O'Reilly Media, Inc.授权中国电力出版社出版

此简体中文版仅限于在中华人民共和国境内（但不允许在中国香港、澳门特别行政区和中国台湾地区）销售发行
This Authorized Edition for sale only in the territory of People's Republic of China (excluding Hong Kong, Macao and Taiwan)

Head First Mobile Web中文版





Lyza Danger Gardner
Jason Grigsby 著
林琪 刘晓兵 等译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'Reilly Media, Inc.授权中国电力出版社出版

图书在版编目（CIP）数据

深入浅出mobile web /（美）加德纳（Gardner,L.D.），（美）格里格斯比（Grigsby,J.）著；
林琪等译. – 北京：中国电力出版社, 2013.3
书名原文: Head first mobile web
ISBN 978-7-5123-4112-8
I. ①深… II. ①加… ②格… ③林… III. ①移动网 IV. ①TN929.5
中国版本图书馆CIP数据核字（2013）第038296号
北京市版权局著作权合同登记
图字：01-2012-2621号

©2011 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and China Electric Power Press, 2013. Authorized translation of the English edition, 2011 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由O'Reilly Media, Inc.出版2011。

简体中文版由中国电力出版社出版，2013。英文原版的翻译得到O'Reilly Media, Inc.的授权。此简体中文版的出版和销售得到出版权和销售权的所有者 —— O'Reilly Media, Inc.的许可。

版权所有，未得书面许可，本书的任何部分和全部不得以任何形式重制。

书　　名/	深入浅出mobile web
书　　号/	ISBN 978-7-5123-4112-8
责任编辑/	刘炽
封面设计/	Karen Montgomery，张健
出版发行/	中国电力出版社
地　　址/	北京市东城区北京站西街19号（邮政编码100005）
印　　刷/	航远印刷有限公司
开　　本/	880毫米×1230毫米　20开本　23.5印张　638千字
版　　次/	2013年6月第1版　2013年6月第1次印刷
印　　数/	0001－3000册
定　　价/	68.00元（册）

敬 告 读 者

本书封底贴有防伪标签，刮开涂层可查询真伪
本书如有印装质量问题，我社发行部负责退换
版 权 专 有　翻 印 必 究

"如果你考虑买一本关于移动开发的书，希望能跨浏览器而且适应不同的开发商，那就赶紧把《Head First Mobile Web（中文版）》收入囊中吧。写这本书的人绝顶聪明，在移动领域有着丰富的经验，而且他们并不满足于一个平台，而是在所有平台上都有历练。很多开发人员可能花很长时间讨论该开发原生应用还是Web应用。这本书由浅入深，以通俗易懂的方式从浅显的入门知识过渡到高级主题，使你轻松掌握创建移动内容所需的全部知识。"

——Andrea Trasatti，DeviceAtlas项目领导人，
无线设备能力信息WURFL存储库的共同创建人

"针对当今这个移动Web开发的纷繁世界，这本书以实效为宗旨为我们娓娓道来，还让我们有机会了解到该如何面向未来。《Head First Mobile Web（中文版）》成功地展示了大量实用技术，所有读者都能立即上手，另外还提供了丰富的基础资源，使有经验的开发人员如虎添翼。"

——Stephen Hay，Web设计人员，开发人员，演讲人，Mobilism会议的共同创办人

"《Head First Mobile Web（中文版）》从一开始就着手实践，对于在Web设计领域探索时可能遇到的机会和挑战做了绝妙的介绍。"

——Bryan 和 Stephanie Rieger，yiibu.com创建人

对其他Head First书的赞誉

"《Head First Object-Oriented Analysis and Design》采用一种全新的视角研究OOAD。这本书独树一帜之处在于它着眼于学习。作者们不仅让实践者能很好地理解OOAD的内容，更重要的是能让他们真正使用这些内容。"

——Ivar Jacobson，Ivar Jacobson Consulting

"我刚刚读完《HF OOA&D》，实在太喜欢这本书了！这本书最让我称道的是它重点讨论了我们为什么要使用OOA&D来编写优秀软件！"

——Kyle Brown，杰出工程师，IBM

"在有趣的图片和夸张的字体后面，是对OO分析与设计认真、睿智的介绍，所有内容都做了精心设计。读这本书时，我感觉就像是站在一个资深设计人员肩上，他一直在为我解释每一步有哪些重要的问题，并且告诉我为什么。"

——Edward Sciore，副教授，波士顿学院计算机科学系，
波士顿学院

"总之一句话，《Head First Software Development》对任何想要提升编程技术的人来说都是一个绝好的资源，它能让各个不同层次的读者都乐于参与其中。"

—— Andy Hudson，Linux Format

"如果你还是一个软件开发新手，《Head First Software Development》能让你有一个正确的起点。如果你是一个有经验的开发人员（在这个领域已经打拼多年），也不要那么急于把它抛在一边……"

—— Thomas Duff，Duffbert's Random Musings

"《Head First Java》适合所有人。不论是视觉学习者还是动觉学习者，每个人都能从这本书中学到东西。视觉效果可以让知识更容易记住，这本书采用了一种非常易读的方式编写，这与大多数Java手册截然不同…… 它是一本极有价值的书。我发现《Head First books》已经用于课堂，这包括高中以及成人考试课程。我肯定还会参考这本书，当然也会告诉别人参考这本书。"

—— Warren Kelly，Blogcritics.org，2006年3月

"不再是一种照本宣科式的学习，《Head First iPhone and iPad Development》提供了一种幽默风趣、生动形象甚至充满乐趣的方式来学习iOS开发。这里涵盖了很多关键技术，包括核心数据甚至诸如接口设计等重要方面，这些内容都经过精心挑选，而且都是最顶尖的新技术。你在哪里能看到UIWebView和UITextField居然能坐在一起亲切交谈！"

—— Sean Murphy，iOS设计师和开发人员

"《Head First Java》第二版还有一个优点，它能满足更多人的口味。现在它涵盖了一些更高级的主题，如Swing和RMI，你可能早已经等不及想要深入了解这些API，用Java.net编写长达十万行代码却无一点瑕疵的程序，这会让你名利双收。书中还提供了有关网络和线程的大量资料，甚至一些最佳实践，这是我的弱项。在这方面，我实在要多夸几句，作者们居然用一个20世纪50年代的电话接线员（嗯，你知道的，就是那种带着耳机人工接线的女生）模拟TCP/IP端口……你真得到书店翻翻《Head First Java》第二版。即使你已经了解Java，也可以从中获得一到两个收获。如果没有，不妨随手翻翻，这也很有意思。"

——Robert Eckstein，Java.sun.com

"当然，并不是《Head First Java》的内容让其独树一帜，它的突出之处在于这本书的风格和方法。利用卡通画、谜题、冰箱磁贴（没错，就是冰箱磁贴），使得这本书绝不是你以前见过的计算机科学教材或技术手册。另外，取代常见的读者练习，这本书要求你假扮成编译器来编译代码，或者可能要通过填空来组织一些代码，或者……你懂的。这本书的第一版是我向初学Java和对象的人推荐的书目之一。这个新版本也没有让我失望，它很好地延续了上一版的优点。如果你像很多人一样，看着一本传统的计算机书就昏昏欲睡，相信这本书肯定能让你一直保持清醒，乐于继续学习。"

—— TechBookReport.com

"《Head First Web Design》是掌握所有这些复杂主题并了解Web设计领域最新进展的钥匙。如果你用过类似Dreamweaver的棘手工具并深受折磨，这本书将是掌握优秀Web设计的绝好途径。"

—— Robert Pritchett，MacCompanion

"还能从书中学习真正的Web设计，这可能吗？《Head First Web Design》就是设计用户友好的网站的法宝，让你对网站开发全过程的设计有所了解，从客户需求，到手绘情节图板，再到正常运行的在线网站。这本书与其他教你'如何构建网站'的书有一个最大的不同，它使用了关于认知科学和学习的最新研究成果，为读者提供了一个包含丰富图片的视觉学习过程，设计这种学习方式就是为了让大脑更好地工作和学习。成果很明显，这将成为Web设计基础的强有力补充，任何通用计算机库都能从中找到成功之路。"

—— Diane C. Donovan，California Bookwatch: The Computer Shelf

"我要向所有涉及艺术方面的程序员同事推荐《Head First Web Design》。"

—— Claron Twitchell，Utah Java用户组

O'Reilly的其他相关图书

- jQuery Cookbook
- jQuery Pocket Reference
- jQuery Mobile
- JavaScript and jQuery: The Missing Manual

O'Reilly的其他 Head First 系列图书

- Head First C#
- Head First Java
- Head First Object-Oriented Analysis and Design (OOA&D)
- Head First HTML with CSS and XHTML
- Head First Design Patterns
- Head First Servlets and JSP
- Head First EJB
- Head First SQL
- Head First Software Development
- Head First JavaScript
- Head First Physics
- Head First Statistics
- Head First Ajax
- Head First Rails
- Head First Algebra
- Head First PHP & MySQL
- Head First PMP
- Head First Web Design
- Head First Networking
- Head First iPhone and iPad Development
- Head First jQuery
- Head First HTML5 Programming

献给我的家族中非凡的女性们：姐姐Maggie、Momula和Fran; 姨妈Catherine; 继母Christie；最重要的，要献给我永远怀念的祖母Pearl，她旺盛的精力和自主精神鼓舞了一代又一代。

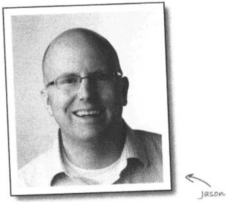
——**Lyza**

感谢我的父母很多年前买了那台Commodore 64；感谢我亲爱的妻子Dana，没有你的支持和理解，就没有这本书；还要感谢Katie和Danny。哈，爸爸终于可以和你们一起玩啦。

——**Jason**

PDG





Lyza Danger Gardner (@lyzadanger)是一位开发人员。她从1996年就在Web领域打拼。Lyza实际上出生在美国俄勒冈的波特兰，并在那里长大，让人奇怪的是，那是一个人人都想去的小镇，但似乎没有哪个名人来自那里。

Lyza很早就上了大学，她的教育经历很杂乱：最初获得了波特兰州立大学艺术和文字文学学士学位，后来又参加了伯明翰大学计算机科学专业的一个硕士研究生计划。

Lyza编写过很多Web应用（主要是服务器端开发），做过漂亮的内容管理系统，对移动网站做过优化，开发过多个API，还参与了很多数据库工作。如今她对移动技术带来的改变相当着迷，正在考虑Web、移动等技术的未来。

从共同创办Cloud Four以来（这是一家总部设在波特兰的移动Web代理机构），2007年，Lyza开始向设备领域进军，着手研究移动浏览器和移动Web的种种问题。她有一大堆奇怪的爱好，据说她拍了相当多的照片。她拥有一个.com域，域名是4个字母。相信你肯定能猜出这是什么，并想去看看。

2000年，**Jason Grigsby**得到了他的第一部手机。他开始考虑如果所有人都能利用他们手中的设备访问信息，这个世界会有怎样的改观。他的妻子Dana遇见他时，他正痴迷于他的移动梦想。直到今天，他还在疑惑她居然嫁给了他。

这些移动梦想被现实打碎了，WAP没有获得成功。所以Jason继续从事Web领域的工作，直到2007年iPhone的问世表明时机终于来了。他同他熟知的3个顶尖人物合作创办了Cloud Four。

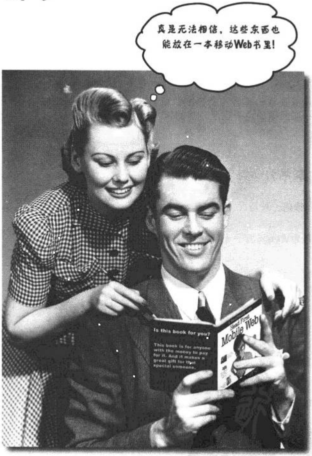
从合作创办Cloud Four以来，他的运气一直不错，参与了很多非常棒的项目，包括Obama iPhone App。他是Mobile Portland的创始人和总裁，这是当地一家非盈利机构，专门致力于在俄勒冈的波特兰推广宣传移动社区。

Jason是移动领域很受欢迎的演讲人和顾问。说实在的，与2000年时相比，他现在对移动领域更为痴迷。

在http://cloudfour.com和他的个人网站http://userfirstweb.com上可以找到他的博客，另外可以在Twitter与他联系（@grigs）。记得要问好哦！

如何使用这本书

引子



有个问题真是听得我们耳朵都磨出茧了：

"你们到底为什么要把这样一些东西放在一本移动Web书里呢？" 现在就来回答这个问题。

谁适合看这本书？

如果对下面的所有问题都能肯定地回答“是”：

① 以前有Web设计或开发经验吗？如果你有编写脚本的经验，这肯定也有帮助。我们并不是在讨论火箭科学，没有那么难，如果你看到一个JavaScript片段，大可不必惊慌失措。

② 你是不是想学习、理解、记住并应用重要的移动Web概念，让你的移动Web页面更有交互性、更让人兴奋？

③ 你是不是更喜欢一种轻松的氛围，就像在晚餐餐桌上交谈一样，而不愿意被动地听枯燥乏味的技术报告？

这正是你要的书。

谁可能不适合看这本书？

如果满足下面任何一种情况：

① 你是不是对Web开发一无所知？

② 你是不是已经在开发移动Web应用或网站，正在找一本关于移动Web的参考书？

③ 你是不是对新鲜事物都畏首畏尾？只喜欢简单的样式，而不敢尝试把条纹和格子混在一起看看？你是不是觉得，如果加入了非凡海象酒吧和一个名叫花格布爱好者的应用，这样一本书肯定不是一本正经八百的技术书？

那么，这本书将不适合你。

[来自市场的声音：只要买得起，不论是用信用卡还是现金，都可以拥有这本书。现金也不错。——小编]



我们知道你在想什么。

"这算一本正经八百的移动Web开发书吗？"

"这些图做什么用？"

"我真能这样学吗？"

我们也知道你的大脑正在想什么。



你的大脑总是渴求一些新奇的东西。它一直在搜寻、审视、期待着不寻常的事情发生。大脑的构造就是如此，正是这一点才让我们不至于墨守成规，能够与时俱进。

我们每天都会遇到许多按部就班的事情，这些事情很普通，对于这样一些例行的事情或者平常的东西，你的大脑又是怎么处理的呢？它的做法很简单，就是不让这些平常的东西妨碍大脑真正的工作。那么什么是大脑真正的工作呢？这就是记住那些确实重要的事情。它不会费心地去记乏味的东西，就好像大脑里有一个筛子，这个筛子会筛掉"显然不重要"的东西，如果遇到的事情枯燥乏味，这些东西就无法通过这个筛子。

那么你的大脑怎么知道到底哪些东西重要呢？打个比方，假如你某一天外出旅行，突然一只大老虎跳到你面前，此时此刻，你的大脑还有身体会做何反应？

神经元会"点火"，情绪爆发，释放出一些化学物质。

好了，这样你的大脑就会知道……

这肯定很重要！可不能忘记了！



不过，假如你正待在家里或者坐在图书馆里，这里很安全、很舒适，肯定没有老虎。你正在刻苦学习，准备应付考试。也可能想学一些比较难的技术，你的老板认为掌握这种技术需要一周时间，最多不超过十天。

这就存在一个问题。你的大脑很想给你帮忙。它会努力地把这些显然不太重要的内容赶走，保证这些东西不去侵占本不算充足的脑力资源。这些资源最好还是用来记住那些确实重要的事情，比如大老虎，再比如遭遇火灾等。就像你的大脑会让你记住绝对不能再穿着短裤去滑雪。

没有一种简单的办法来告诉大脑："嘿，大脑，真是谢谢你了，不过不管这本书多没意思，也不管现在我对它多么无动于衷，但我确实希望你能把这些东西记下来。"

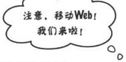
我们认为“Head First”的读者就是要学习的人。

那么，怎么学习呢？首先必须获得知识，然后保证自己确实不会忘记。这可不是填鸭式的硬塞。根据认知科学、神经生物学和教育心理学的最新研究结果，学习的途径相当丰富，绝非只是通过书本上的文字。我们很清楚怎么让你的大脑兴奋起来。

下面是一些Head First学习原则：



看得到。与单纯的文字相比，图片更能让人记得住，通过图片，学习效率会更高（对于记忆和传递型的学习，甚至能有多达89%的效率提升）。而且图片更能让人看懂。以往总是把图片放在一页的最下面，甚至放在另外的一页上，与此不同，如果把文字放在与之相关的图片内部，或者在图片的周围写上相关文字，学习者的能力就能得到多至两倍的提高，从而能更好地解决有关的问题。





采用一种针对个人的交谈式风格。最新的研究表明，如果学习过程中采用一种第一人称的交谈方式直接向读者讲述有关内容，而不是用一种干巴巴的语调介绍，那么学生在学习之后的考试中成绩会提高40%。正确的做法是讲故事，而不是做报告。要用通俗的语言。另外不要太严肃。如果你面对着这样两个人，一个是你在餐会上结识的很有意思的朋友，另一个人学究气十足，喋喋不休地对你说教，在这两个人中，你会更注意哪一个呢？

让学习的人想得更深。换句话说，除非你很积极地让神经元活动起来，否则你的头脑里什么也不会发生。必须引起读者的好奇，促进、要求并鼓励读者去解决问题、得出结论、产生新的知识。为此，需要发出挑战，留下练习题和拓宽思路的问题，并要求读者完成一些实践活动，让左右脑都开动起来，而且要利用到多种思维。



引起读者的注意，而且要让他一直保持注意。我们可能都有过这样的体验，“我真的想把这个学会，不过看过一页后实在是让我昏昏欲睡”。你的大脑注意的是那些不一般、有意思、有些奇怪、抢眼的、意料之外的东西。学习一项有难度的新技术并不一定枯燥。如果学习过程不乏味，你的大脑很快就能学会。

影响读者的情绪。现在我们知道了，记忆能力很大程度上取决于所记的内容对我们的情绪有怎样的影响。如果是你关心的东西，就肯定记得住。如果让你感受到了什么，这些东西就会留在你的脑海中。不过，我们所说的可不是什么关于男孩与狗的伤心故事。这里所说的情绪是惊讶、好奇、觉得有趣、想知道“什么……”还有就是一种自豪感，如果你解决了一个难题，学会了所有人都觉得很难的东西，或者发现你了解的一些知识竟是那些自以为无所不能的傲慢家伙所不知道的，此时就会有一种自豪感油然而生。

元认知：有关思考的思考

如果你真的想学，而且想学得更快、更深，就应该注意你怎样才会专注起来，考虑自己是怎样思考的，并了解你的学习方法。

我们中间大多数人长这么大可能都没有上过有关元认知或学习理论的课程。我们想学习，但是很少有人教我们怎么来学习。

不过,这里可以做一个假设，如果你手上有这本书，你想学习项目管理，而且可能不想花太多时间。如果你想把这本书中读到的知识真正用起来，就需要记住你读到的所有内容。为此，必须理解这些内容。要想最大限度地利用这本书或其他任何一本书，或者掌握学习经验，就要让你的大脑负起责来，要求它记住这些内容。



怎么做到呢？技巧就在于要让你的大脑认为你学习的新东西确实很重要，对你的生活有很大影响。就像老虎出现在面前一样。如若不然，你将陷入旷日持久的拉锯战中，虽然你很想记住所学的新内容，但是你的大脑却会竭尽全力地把它们拒之门外。

那么究竟怎样才能让大脑把移动Web开发看作是一只饥饿的老虎呢？

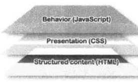
这有两条路，一条比较慢，很乏味。另一条路不仅更快，还更有效。慢方法就是大量地重复。你肯定知道，如果反反复复地看到同一个东西，即便再没有意思，你也能学会并记住。如果做了足够的重复，你的大脑就会说，“尽管看上去这对他来说好像不重要，不过，既然他这样一而再、再而三地看同一个东西，所以我假定这应该是重要的。”

更快的方法是尽一切可能让大脑活动起来，特别是开动大脑来完成不同类型的活动。如何做到这一点呢？上一页列出的学习原则正是一些主要的可取做法，而且经证实，它们确实有助于让你的大脑全力以赴。例如，研究表明，把文字放在所描述图片的中间（而不是放在这一页的别处，比如作为标题，或者放在正文中），这样会让你的大脑更多地考虑这些文字与图片之间有什么关系，而这就会让更多的神经元点火。让更多的神经元点火 = 你的大脑更有可能认为这些内容值得关注，而且很可能需要记下来。

交谈式风格也很有帮助，当人们意识到自己在与“别人”交谈时，往往会更专心，这是因为他们总想跟上谈话的思路，并能做出适当的发言。让人惊奇的是，大脑并不关心“交谈”的对象究竟是谁，即使你只是与一本书“交谈”，它也不会在乎！另一方面，如果写作风格很正统、干巴巴的，你的大脑就会觉得，这就像坐在一群人当中被动地听人做报告一样，很没意思，所以不必在意对方说的是什么，甚至可以打瞌睡。

不过，图片和交谈风格还只是开始而已，能做的还有很多。

我们是这么做的：



我们用了很多图，因为你的大脑更能接受看得见的东西，而不是纯文字。对你的大脑来说，一图胜千言。如果既有文字又有图片，我们就会把文字放在图片当中，因为文字处在所描述的图片中间时，大脑的工作效率更高，倘若把这些描述文字作为标题，或者“淹没”在别处的大段文字中，就达不到这种效果了。

我们采用了重复手法，会用不同方式，采用不同类型的媒体，运用多种思维手段来介绍同一个东西，目的是让有关内容更有可能储存在你的大脑中，而且在大脑中多个区域都有容身之地。



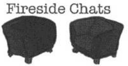
我们会用你想不到的方式运用概念和图片，因为你的大脑喜欢新鲜玩艺。在提供图和思想时，至少会含着一些情绪因素，因为如果能产生情绪反应，你的大脑就会投入更大的注意。而这会让你感觉到这些东西更有可能被记住，其实这种感觉可能只是有点幽默，让人奇怪或者比较感兴趣而已。

我们采用了一种针对个人的交谈式风格，因为当你的大脑认为你在参与一个会谈，而不是被动地听一场演示汇报时，它就会更加关注。即使你实际上在读一本书，也就是说在与书“交谈”，而不是真正与人交谈，但这对你的大脑来说并没有什么分别。



在这本书里，我们加入了大量实践活动，因为与单纯的阅读相比，如果能实际做点什么，则你的大脑会更乐于学习，更愿意去记。这些练习都是我们精心设计的，有一定的难度，但是确实能做出来，因为这是大多数人所希望的。

我们采用了多种学习模式，因为尽管你可能想循序渐进地学习，但是其他人可能希望先对整体有一个全面的认识，另外可能还有人只是想看一个例子。不过，不管你想怎么学，要是同样的内容能以多种方式来表述，这对每一个人都会有好处。



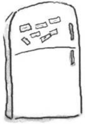
这里的内容不只是单单涉及左脑，也不只是让右脑有所动作，我们会让你的左右脑都开动起来，因为你的大脑参与得越多，你就越有可能学会并记住，而且能更长时间地保持注意力。如果只有一半大脑在工作，通常意味着另一半有机会休息，这样你就能更有效率地学习更长时间。

我们会讲故事，留练习，从多种不同的角度来看同一个问题，这是因为，如果要求大脑做一些评价和判断，它就能更深入地学习。



我们会给出一些练习，还会问一些问题，这些问题往往没有直截了当的答案，通过克服这些挑战，你就能学得更好，因为让大脑真正做点什么的话，它就更能学会并记住。想想吧，如果只是在体育馆里看着别人流汗，则对于保持你自己的体形肯定不会有什么帮助，正所谓临渊羡鱼，不如退而结网。不过另一方面，我们会竭尽所能不让你钻牛角尖，把劲用错了地方，而是能把功夫用在点子上。也就是说，你不会为弄明白一个难懂的例子而耽搁，也不会花太多时间去弄明白一段艰涩难懂而且通篇行话的文字，我们的描述也不会太过简洁而让人无从下手。

我们用了拟人手法。在故事中，在例子中，还有在图中，你都会看到人的出现，这是因为你本身是一个人，不错，这就是原因。如果和人打交道，相对于某件东西而言，则你的大脑会更为关注。



可以用下面的方法让你的大脑就范

好了，我们该做的已经做了，剩下的就要看你自己的了。以下提示可以作为一个起点：听一听你的大脑是怎么说的，弄清楚对你来说哪些做法可行，哪些做法不能奏效。要尝试新鲜事物。

把这一页撕下来，贴到你的冰箱上。

---

① 慢一点。你理解的越多，需要记的就越少。

不要光是看看就行了。停下来，好好想一想。书中提出问题的时候，你不要直接去翻答案。可以假想真的有人在问你这个问题。你让大脑想得越深入，就越有可能学会并记住它。

② 做练习，自己记笔记。

我们留了练习，但是如果这些练习的解答也由我们一手包办，那和有人替你参加考试有什么分别？不要只是坐在那里看着练习发呆。拿出笔来，写一写画一画。大量研究都证实，学习过程中如果能实际动动手，这将改善你的学习。

③ 阅读“没有傻问题”。

顾名思义。这些问题不是可有可无的旁注，它们绝对是核心内容的一部分！千万不要跳过去不看。

④ 上床睡觉之前不要再看别的书，至少不要看其他有难度的东西。

学习中有一部分是在你合上书之后完成的（特别是，要把学到的知识长久地记住，这往往无法在看书的过程中做到）。你的大脑也需要有自己的时间，这样才能再做一些处理。如果在这段处理时间内你又往大脑里灌输了新的知识，那么你刚才学的一些东西就会丢掉。

⑤ 要喝水，而且要多喝点水。

能提供充足的液体，你的大脑才能有最佳表现。如果缺水（可能在你感觉到口渴之前就已经缺水了），学习能力就会下降。

⑥ 讲出来，而且要大声讲出来。

说话可以刺激大脑的另一部分。如果你想看懂什么，或者想更牢地记住它，就要大声地说出来。更好的办法是，大声地解释给别人听。这样你会学得更快，而且可能会有以前光看不说时不曾有的新发现。

⑦ 听听你的大脑怎么说。

注意一下你的大脑是不是负荷太重了。如果发现自己开始浮光掠影地翻看，或者刚看的东西就忘记了，这说明你该休息一会儿了。达到某个临界点时，如果还是一味地向大脑里塞，这对于加快学习速度根本没有帮助，甚至还可能影响正常的学习进程。

⑧ 要有点感觉。

你的大脑需要知道这是很重要的东西。要真正融入到书中的故事里。为书里的照片加上你自己的图题。你可能觉得一个笑话很蹩脚，不太让人满意，但这总比根本无动于衷要好。

⑨ 真正做些工作！

把这些知识应用到你的日常工作中去；利用你所学的想法完成项目决策。具体做些工作，你会得到书中练习和活动以外的经验。所需要的只是一支笔和一个要解决的问题……通过使用针对考试所学的工具和技术可以更好地解决这个问题。

重要说明

要把这看作是一个学习过程，不要简单地把它看成是一本参考书。我们在安排内容的时候有意做了一些删减，只要是对有关内容的学习有妨碍，我们都毫不留情地一律删掉。另外，第一次看这本书的时候，要从第一页从头看起，因为书中后面的部分会假定你已经看过而且学会了前面的内容。

希望你了解HTML和CSS。

如果你不了解HTML和CSS，读这本书之前请先找一本《Head First HTML with CSS & XHTML》看看。我们会稍稍复习比较麻烦的CSS选择器或HTML元素，不过不要指望在这里学习基础知识。

希望你熟悉Web脚本代码。

我们并不要求你是世界级的JavaScript专家，也不要求你之前使用PHP完成过学位水平的计算机科学项目，不过在这本书中你会看到很多用这两种语言完成的例子。如果一个for循环都让你紧张不已（或者如果你根本不知道我们在讲些什么），可以找一本《Head First PHP & MySQL》或《Head First JavaScript》看看，然后再来看这本书。

希望你知道如何查找和记录。

坦率地讲，移动Web是一个覆盖面很广的主题，要想掌握移动Web，你现有的Web开发技能还不够，需要进一步扩展。关于Web需要知道的东西太多了，任何一个人都没有办法完全记住，不论是JavaScript语法细节，还是对某个HTML5元素属性的浏览器支持的具体细节，统统都记住不太可能。不要太难为自己。作为一个好的Web开发人员，在你的众多开发利器中，一定要把Google这把大斧磨锋利，知道什么时候以及如何在网上搜寻Web的有关信息。相信你在这方面已经是高手了。

不要仅限于这本书。

我们要面对一个博大而美丽的移动Web世界。希望我们能扶你上路，顺利地开始你的旅程，不过师父领进门，修行靠个人。移动Web群体相当活跃，你要参与这个群体的在线交流，阅读相关的博客，加入感兴趣的邮件列表，积极地参加这个领域的技术活动。

书里的实践活动绝不是可有可无的。

这里的练习和实践活动不是可有可无的装饰和摆设，它们也是这本书核心内容的一部分。其中有些练习和活动有助于记忆，有些能够帮助你理解，还有一些对于如何应用所学的知识很有帮助。千万不要把这些练习跳过不做。甚至填字游戏也很重要，它们可以帮助你把有关概念植入你的大脑。不过更重要的是，它们可以让你的大脑有机会在不同的上下文思考这些词汇和概念。

我们有意安排了许多重复，这些重复非常重要。

Head First系列图书有一个与众不同的地方，这就是，我们希望你确确实实地掌握这些知识，另外希望在学完这本书之后你能记住学过了什么。大多数参考书都不太重视重复和回顾，但是由于这是一本有关学习的书，你会看到一些概念一而再、再而三地出现。

“Brain Power（头脑风暴）”练习没有答案。

有一些头脑风暴练习根本没有正确的答案，另外一些练习中，头脑风暴实践活动的一部分学习过程就是让你确定你的答案是否正确，以及在何种情况下正确。有些头脑风暴练习中，你会得到一些提示，为你指明正确的方向。

软件需求

与开发所有网站一样，你需要一个文本编辑器、一个浏览器、一个Web服务器（可以本地安装在你的个人计算机上），另外还要有本书各章示例应用的源代码。

对于Windows，我们推荐的文本编辑器是PSPad、TextPad或EditPlus（不过，如果别无选择，也可以使用Notepad）。对于Mac，我们推荐的文本编辑器是TextWrangler（或者它的同系列工具BBEdit），也可以使用TextMate。我们也很喜欢Coda，这是一个更关注Web的工具。

如果你使用的是Linux系统，系统中已经内置了很多文本编辑器，那么相信不需要我们多说了。

如果你要做Web开发，需要一个Web服务器。先要看看附录ii，了解如何安装支持PHP的Web服务器。建议你现在就完成这个工作。说真的，现在就动手，按照操作说明安装开发环境，然后再回来看这一页。

对于第5章，你需要安装WURFL（无线通用资源文件）API和数据。对于第8章，你要安装Android SDK和一些相关的工具。你肯定猜到了,分别有相应的附录教你完成这些任务。

你还需要一个浏览器，噢，应该说你需要尽可能多的浏览器来完成测试。你手上的移动设备（带浏览器）越多越好（别害怕。如果没有硬件，那么还有很多模拟器可供你使用）。

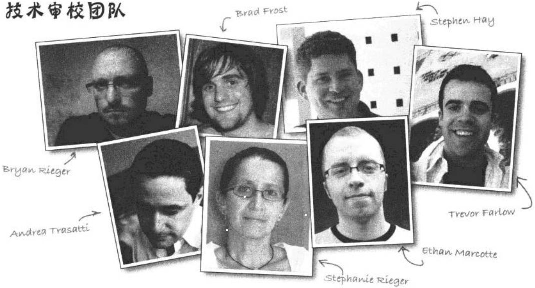
对于桌面开发和测试，我们强烈推荐Google的Chrome浏览器，它在Mac、Windows和Linux平台上分别有相应的版本。应该花点功夫了解如何在Google的Chrome Dev Tools中使用JavaScript控制台，这绝对是值得的。这个任务作为家庭作业需要你自己完成。

最后一点，你要得到各章示例的代码和资源。可以在http://hf-mw.com上找到。

hf-mw.com网站提供了各章的代码。可以到这个网站上下载示例代码。

各章中例子的代码和资源都可以在http://hf-mw.com上找到。

技术审校团队



Trevor Farlow是一位业余面包师，作为消遣喜欢踢足球，另外还兼职做动物保护志愿者。如果没在遛狗、踢足球或者精心烤制他的纽约风格乳酪蛋糕，可能就在Clearwater Analytics, LLC的一个精简高效的敏捷开发小组中学习产品所有权技巧。

Brad Frost是一个移动Web策略师，另外是纽约R/GA的一名前端开发人员，在这里他参与完成了大量与移动相关的项目。他运营着一个资源网站，名为Mobile Web Best Practices (http://mobilewebbestpractices.com)，着力帮助人们创建完美的移动Web体验。

Stephen Hay已经有超过16年构建网站的经验。除了客户要求的工作外（主要是多平台设计和开发），他还在很多行业会议上演讲，并为很多出版商撰稿，如A List Apart和.net Magazine。他还合作组织了Mobilism，这是一个很受推崇的移动Web设计和开发会议。

Ethan Marcotte是一位独立设计人员/开发人员，他对漂亮的设计、精巧的代码和这二者的结合相当痴迷。多年来，他积累了大量客户，包括New York Magazine、Sundance Film Festival、Boston Globe和W3C。Ethan首创了响应式Web设计一词，来描述为不断变化的Web进行设计的一种新方法，如果有机会，则他甚至想关于这个主题写一本书。

Bryan Rieger是一名设计人员和敬业的开发人员，拥有剧场设计和经典动画制作背景。Bryan已经从事过多种媒体行业，包括印刷、广播、Web和移动。他的客户包括Apple、Microsoft、Nokia和Symbian Foundation。Bryan很热爱讲故事，总在修修补补，你会看到他在Yiibu做各种各样的工作，这是一家很小的设计顾问机构，总部设在苏格兰的爱丁堡。

Stephanie Rieger是一位设计人员、作家和人类学家，非常关心人们与技术交互的多种途径。Stephanie从2004年就在从事移动设计工作，现在主要关注Web策略、前端设计以及针对多屏幕和能力的优化。作为一个很有干劲的测试人员和研究人员，Stephanie总是热衷于发现和分享关于移动用法、用户行为和全世界移动领域趋势的见解。

Andrea Trasatti 1999年在Nokia 7110上开始创建WAP内容，那时在欧洲这被认为是开创性的技术。Andrea领导着WURFL和DeviceAtlas从最初起步直到最后取得成功，在此期间，他在设备检测和内容适应方面取得了丰富成果。可以在Twitter上找到Andrea（@AndreaTrasatti），他经常在谈论移动Web以及创建和管理移动内容的新趋势。

致谢



Courtney Nash

致编辑：

感谢（同时祝贺）Courtney Nash，在她的推动下，我们终于完成了这本书，也许这是我们最棒的一本书。长期以来，她一直在忍受着我们的大量邮件、问题、长篇大论、有时甚至还要接受我们的偏执。她坚持带领我们完成了这本书，一直相信我们有信心克服所有困难。另外要感谢Brian Sawyer最后的督促，带着我们冲向终点。

致O'Reilly人员：



Lou Barr

感谢Lou Barr无与伦比的高效以及在设计和布局方面的高超水平。我们真的叹为观止。谢谢你还要感谢Karen Shaner和Rachel Monaghan，谢谢你们帮助我们修改草稿，完成全书的审校，还帮我们解决方方面面的细节问题！

感谢O'Reilly的其余同仁，让我们感觉到家庭般温暖：Mike Hendrickson，感谢你刚开始建议了这个绝妙的想法；Brady Forrest，感谢你介绍并一直支持我们；Tim O'Reilly，你真是一个睿智、诚恳的好人；还有Sara Winge，感谢你的恩威并重。

还要感谢：

Jason和Lyza有幸能在Cloud Four与一群最睿智的人一同工作。我们要特别感谢合作创办人Aileen Jeffries和John Keith，还有Cloud Four团队的其他人员：Matt Gifford、Chris Higgins和Megan Notarte。这本书实际上是我们在移动Web领域的共同结晶。他们提供了最大的支持，付出了最大的努力。非常非常感谢你们！

还要感谢移动Web社区。特别地，我们要感谢Josh Clark、Gail Rahn Frederick、Scott Jehl、Scott Jenson、Dave Johnson、Tim Kadlec、Jeremy Keith、Peter Paul Koch、Brian LeRoux、James Pearce、Steve Souders和Luke Wroblewski。能成为这个社区的一分子我们很骄傲，也心存感激。

Lyza的朋友和家人：

感谢Bryan Christopher Fox（Other Dev），没有你修改代码，没有你的真知灼见、鼎力支持和超强智慧，这本书绝无可能出版。

感谢我的朋友和家人，因为写书，我总是无法和你们在一起，谢谢你们能够忍受这段漫长时光。感谢Autumn和Amye，在看不到我的时候你们表现出了非凡的忍耐力。谢谢你，Mike，永远感谢你。还要感谢亲爱的爸爸，是你总是教导我要发现美和新的冒险。最后，感谢威尔士Llandegla Plas-yn-Iâl的Huw和Bethan，感谢你们提供了这样一个绝妙的地方让我们写书。

Jason的朋友和家人：

感谢我的家人给予的所有支持。爸爸妈妈、Jan、Carol、Mark和Doanne，是你们一如既往的帮助，让我们在写书、照顾家里和搬家的艰难历程中一直保持心平气和。

特别感谢我的妻子，Dana Grigsby，要知道，在我写书期间，我们要照顾一个婴儿、一个学龄前孩子，还要搬新家，没有你我真的做不到。

Safari®图书在线

Safari图书在线是一个按需提供资源的数字图书馆，从中可以很容易地搜索7500本技术书、参考书和视频，快速找到你想要的答案。

只需订阅，就可以从我们的在线图书馆阅读任何页面，观看任何视频。你可以在你的手机和移动设备上看书，能够在新书出版之前获得书目，还可以享有特权查看正在编写的书稿并向作者提出反馈意见。你可以剪切粘贴代码示例、整理最喜欢的图书、下载你需要的章节、为重要内容设置书签、创建笔记、打印页面，此外还有大量特性可以节省时间，让你从中受益。

O'Reilly Media已经将这本书上传到Safari图书在线服务。要想通过数字方式全面访问这本书以及O'Reilly和其他出版商提供的其他类似图书，可以免费注册http://my.safaribooksonline.com。

目录（概览）

	引子	xxi
1	移动Web入门：响应式Web设计	1
2	真正的响应性：移动优先响应式Web设计	43
3	单独的移动网站：直面不太好的环境	91
4	决定支持谁：我们要支持哪些设备？	137
5	设备数据库和类：分组行动	151
6	使用框架构建移动Web应用：花格布爱好者	217
7	真实世界的移动Web应用：超级移动Web应用	267
8	用PhoneGap构建混合型移动Web应用：花格布搜寻：实现原生	313
9	如何实现未来友好：应对混乱	357
i	其他：6大主题（我们没有谈到的）	373
ii	建立Web服务器环境：就要开始了	387
iii	安装WURFL：搜寻设备	397
iv	安装Android SDK和工具：完善环境	403
	索引	417

详细目录

引子

你的大脑与移动Web。你想学些新东西，但是你的大脑总是帮倒忙，它会努力让你记不住所学的东西。你的大脑在想："最好留出空间来记住那些确实重要的事情，比如要避开哪个野生动物，还有裸体滑雪是不是不太好。"那么，如何让你的大脑就范？让它认为如果不知道移动Web你将无法生存！

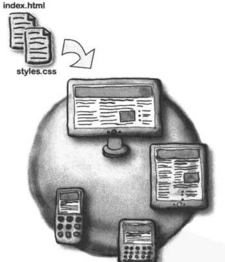
谁适合看这本书？	xxii
我们知道你在想什么	xxiii
我们也知道你的大脑正在想什么	xxiii
元认知：有关思考的思考	xxv
技术审校团队	xxx
致谢	xxxi

1 移动Web入门

响应式Web设计

嘿！准备好了吗？要进入移动世界吗？

移动Web开发绝对是一种让人惊心动魄的生活方式。它有十足的魔力，让人兴奋不已，欲罢不能，你会一次次地感受到顿悟的快乐！不过，这里也有谜团，也有困惑。移动技术正在飞速发展，需要了解的东西实在太多了！快抓紧，就要开始我们的移动之旅了，我们会带着你了解一种建设网站的方法，称作响应式Web设计（Responsive Web Design，RWD）。采用这种方法，可以利用你已经掌握的Web技能适当地调整网站，让它们在各种移动设备上同样有非凡表现。



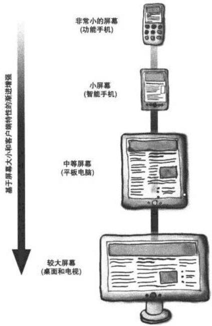
登上“移动”花车	2
酒吧发生了怪事	4
如果移动手机Web浏览器这么棒……	5
移动Web为什么有这么大差异？	6
响应式Web设计	10
不同场合使用不同的CSS	12
CSS媒体查询	13
当前非凡海象网站的结构	15
分析当前CSS	16
哪些需要改变？	17
找出需要修改的CSS	18
创建移动CSS的步骤	19
固定宽度布局到底有什么问题？	26
为什么流式网格布局更好？	27
流公式	28
继续流式转换	29
上下文切换	31
这个图片怎么了？	32
流式图像和媒体	33
记住你的责任	36
得到一个响应式网站！	40
响应式设计也是一种精神状态	41

2 真正的响应性

移动优先响应式Web设计

这是一个漂亮的移动网站。不过漂亮只是表面。

在内部，它根本不是那样。它可能看上去像是一个移动网站，不过仍然是一个披着移动外衣的桌面网站。如果我们希望这个网站成为移动世界里一道炫目的闪电，就要先从移动开始。首先分解当前的网站，找出隐藏在移动外衣下的桌面成分。我们会做个大扫除，从头开始改头换面，从构建基本内容直到建立一个桌面视图。完成后，你会得到一个不论屏幕大小都可以优化的页面。



你以为该庆祝胜利了……	44
这算问题吗？我们怎么知道？	45
速度不快时怎么办	47
不要被外表迷惑，这实际上是一个很大的页面	48
HAR山里有黄金	49
找出影响页面速度的累赘	51
Google Maps JavaScript从哪里来？	53
看起来是移动友好的，但实际上不是	55
移动优先响应式Web设计	56
什么是渐进增强？	57
修正内容浮动	60
移动优先媒体查询	61
意外！Internet Explorer中页面出问题了	62
一个src完全控制	68
在viewport <meta>标记上控制放大	72
放大的权利？	73
使用JavaScript增加地图	74
在JavaScript中建立一个伪媒体查询	76
将JavaScript增加到On Tap Now页面	77
这些部件不是响应式的	79
将iframe属性移至CSS	80
从JavaScript删除属性	81
又出现地图重叠了	83
让内容成为你的向导	84
分界点来帮忙	87

3 单独的移动网站

直面不太好的环境

响应式Web的远景固然美妙……

在这里，每个网站都采用一种移动优先的方法巧妙构建，只有一个布局，可以应对各种设备和环境。嗯……真不错。不过，如果现实一点，考虑一些实际因素会怎么样呢？比如说，遗留的系统、老式的设备，另外还有可能受到顾客预算的限制。有些时候，如果实在不能把桌面和移动支持完美地结合在一起，是不是需要单独来建设？这一章中，我们会介绍有关的细节问题，包括检测移动用户，支持那些老式手机，以及构建一个单独的移动网站。



Creature Comforts的现场代理处	92
代理处如何得到和分享他们需要的信息？	93
让移动用户前往移动优化网站	96
寻找移动用户	97
认识用户代理	98
用户代理：魔鬼蛋	101
实话实说：大多数主要网站都有一个单独的移动网站	104
如果你真正想要的是重定向	105
看一眼脚本	106
这个脚本如何工作？	107
建立一个移动模拟网站	108
复杂因素……的特别快递	110
并不是所有手机都是智能手机……完全不是这样	113
力求基本：认识XHTML-MP	114
为什么我们要用这个旧东西？	115
XHTML-MP力求简单	116
顺便说一句，还有滚动问题	119
最后一个问题	119
访问键的实际使用	123
怎么回事？	124
修正错误	125
懂移动的CSS	127
嗯……少了点东西	132
按钮外观没有了！	133
大获成功！	134

4 决定支持谁

我们要支持哪些设备？

我们实在没有那么多时间在每一个设备上测试。

你必须划分出一个界线，确定能支持哪些设备。不过如何确定呢？有些人可能使用一些特殊的设备，而你无法在那些设备上测试，对于这种情况你该如何处理？完全不考虑他们吗？或者有没有可能采用一种合适的方式构建Web页面，使人们用你从未听说过的设备也能访问？这一章中，我们将结合项目的需求和用户的使用情况制作一种魔法剂，帮助我们确定支持哪些设备，以及对于不支持的设备该如何处理。

你怎么知道在哪里划分界线？	138
稍事休息	139
不支持的设备与不能支持的设备	140
关于你的项目问几个问题	142
移动魔法剂的配方	144
利用工具和数据	145
我怎么知道顾客有合适的设备呢？	150



5 设备数据库和类

分组行动

尽管能确定我们支持的设备，但有些烦人的小问题还不能解决。

如何了解用户移动浏览器足够的信息，在向他们发布内容之前能够知道这些浏览器确实符合要求？如何避免只为那些最低档次的设备构建内容（往往很简陋）？另外如何合理组织所有内容，以免我们焦头烂额？这一章中，我们将进入设备能力的领地，学习如何用设备数据库访问设备能力，最后还会介绍如何将它们归组为设备类，使我们能安然应对。







学生有麻烦，提供紧急按钮 152
移动设备数据源可以帮忙 154
来认识认识WURFL 155
WURFL及其能力 156
WURFL: 聪明的API代码 159
我们也可以建立一个资源管理页面 160
资源管理页面：建立我们的环境 161
完成一个快速的双连击，改进我们的资源管理页面 168
把这些能力信息派上用场 170
使用WURFL帮助区分内容 170
初始化设备，准备好信息 172
它是移动的吗？ 172
利用WURFL让页面更聪明一些 176
紧急按钮：只针对手机 177
设备类 181
扩大AcedIt!的业务范围 182
戴上移动眼镜评判这个主页 183
将需求分组为多种移动体验 184
设备类排排坐 185
熟悉匹配函数 191
这个switch语句在做什么？ 192
使用匹配函数测试能力 193
填补设备类测试的缺口 200
我们需要一个更大的安全网 211
及时的一针 212



6 使用框架构建移动Web应用

花格布爱好者

"我们想要个应用！"

最近一到两年，我们常常听到这样的呼吁，这通常意味着要为你想要支持的每一个平台完成原生代码开发和部署。不过，原生应用不再独占江山。如今，面向移动浏览器的Web应用已经蜂拥而至，特别是现在HTML5和他的死党CSS3和JavaScript都在摩拳擦掌、跃跃欲试。下面就让我们利用一个移动框架更深入地探寻这个移动Web应用世界，移动框架就是专门设计用来帮助你快速完成工作的代码工具！



HTML5……应用……这些词究竟是什么意思？ 219
"传统"网站的一般表现 220
应用型网站的一般表现 221
Tartanator第1阶段主计划 224
为什么使用移动Web应用框架？ 225
Tartanator的选择：jQuery Mobile 226
用jQuery Mobile建立一个基本页面 228
页面其余的标记 229
HTML5 data-*属性 231
用jQuery Mobile链接到多个页面 234
让列表麻雀变凤凰 241
加入其余的花格布 243
过滤和组织列表 244
增加一个页脚工具栏 249
让工具栏更时尚 250
最终确定结构 251
建立花格布创作表单 253
把花格布图案转换成一个表单 255
构建一个HTML5表单 256
增加一些基本的域 257
嵌套列表允许用户增加颜色 258
颜色-大小域对：颜色选择域 259
颜色-大小域对：大小域 260
链接到表单 262

7 真实世界的移动Web应用

超级移动Web应用

移动Web就像班里那个小机灵鬼。

你知道的，就是那种很让人喜欢、聪明能干的小家伙，不过也是捉摸不定、让人防不胜防的淘气包。我们尽量不设置太多约束和条条框框，希望不扼杀他们过人的天赋，不过现在该是有效利用移动Web天分的时候了。利用渐进增强，我们可以在更成熟的浏览器中把原本简朴的界面装扮一新。通过建立一种周全的离线模式，能够让飘忽不定的网络连接从一种负担转变成一个抢眼的特性。另外，通过使用地理定位可以得到移动的真谛。下面就来建立这样一个超级移动Web应用吧！



看着还不错……	268
真实世界的移动应用	270
各就各位，预备，增强！	274
建立一个更棒的表单	275
一个部件管理颜色大小列表	276
深度探秘	277
这就是前端增强……	278
……现在来看后端	280
generate.php的两方面	281
最后一个问题！	282
离线很重要	284
创建缓存清单的基本步骤	285
开发工具来帮忙	286
提供内容类型正确的清单	287
胜利（最终）属于我们	297
地理定位如何工作	298
如何向符合W3C标准的浏览器询问当前位置	299
查找事件页面起步：基准	301
集成地理定位	303
什么也没找到	309

8 用PhoneGap构建混合型移动Web应用

花格布搜寻：实现原生

有时候你得实现原生。

这可能是因为你需要访问移动浏览器（还）没有提供的某些功能。或者，也许是你的客户要求有App Store中的一个应用。我们期盼着将来能够在浏览器中访问我们想要的一切，希望移动Web应用同样拥有那些诱人的原生应用才拥有的特性。这种情况下，我们可以选择混合型开发，继续使用Web标准编写代码，再使用一个库在我们的代码与设备原生能力之间搭建桥梁。基于Web技术构建的跨平台原生应用？不会是一个糟糕的折中吧，嗯？



机会又来了	314
混合型应用如何工作？	317
用PhoneGap填补Web-原生缺口	318
熟悉PhoneGap Build	321
应用如何工作？	322
跟踪已经找到的花格布	323
Tartan Hunt项目剖析	324
下载应用	328
选择你的路	329
看到了什么？存储找到的花格布	334
localStorage能为我们做什么？	335
检查浏览器支持什么	339
用一个函数显示哪些花格布已经找到	340
toggle和toggleClass方法	341
你找到了一个花格布，是吗？那就证明看看！	344
利用PhoneGap拍照	345
PhoneGap基本准备就绪，可以出场了	347
现在准备使用mediaCapture API	348
成功时如何处理？	349
真实世界里总有点不同	350
只是匿名	351
最后一点！	353
搞定了！	354

9 如何实现未来友好

应对混乱

响应式Web设计、设备检测、移动Web应用、PhoneGap。等一下……我们该用哪一个呢？

移动Web开发方法实在太多了。通常，项目中都会结合应用多种技术。这方面并没有唯一的正确答案。不过不用担心。关键是要学会“随波逐流”。掌握不确定性。用一种未来友好的思维，顺应潮流，确保一定要灵活，能够适应未来的一切变化。

现在怎么样呢？ 358

消除控制假象 361

未来友好宣言 362

如果你不能保证未来，那就实现未来友好吧 364

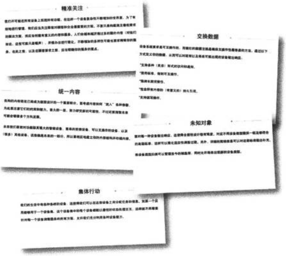
今天的应用，明天的网页 365

漫长的道路：这里有一些路标 366

移动大餐 369

面向未来 371





其他

i

6大主题（我们没有谈到的）

是不是感觉少了点什么？我们知道你是什么意思……

你以为已经完成了，但实际上还有一些事情没有做。如果不再做一点补充（这些内容不适合放在前面的正文里），我们实在不忍心就此放手。至少如果你还没有一个带轮子的金属旅行箱，就想带着这本书闯荡世界，那是肯定不行的。来看看（还）少哪些内容。

#1 在移动设备上测试 374

#2 远程调试 376

#3 确定浏览器支持什么 382

#4 设备API 384

#5 应用商店和分发 385

#6 RESS: 响应式设计 + 服务器端组件 386

建立Web服务器环境

ii

就要开始了

没有“Web”，“移动Web”无从谈起。

对于这一点绝对没有异议。如果要完成移动Web开发，肯定需要有一个Web服务器。这远不只是完成本书中的练习那么简单。你要把你的Web托管内容放在某个地方，不论是使用一个第三方商业Web托管服务，还是企业级数据中心，或者甚至是你自己的计算机。在这个附录中，我们将带领你在你的计算机上逐步建立一个本地Web服务器，并使用免费开源的软件运行PHP。

你要做的准备 388

只在本地可用 389

Windows和Linux：安装和配置XAMPP 390

启动XAMPP 391

Mac用户：现在是MAMP时间 392

确保有正确的端口 393

访问你的Web服务器 394

phpInfo，拜托了！ 396

iii

安装WURFL

搜寻设备

要揭开设备检测的神秘面纱，第一步要做些调查工作。

优秀的侦探都知道，要解开谜团，必须搜集线索，询问目击证人。首先，我们要找出行动的策划者：WURFL PHP API。接下来，追踪行动者：将数以千计设备的能力信息放在一个XML数据文件中。不过还需要做些协调，才能让它们协作解决整件事情，所以我们要对配置稍做调整，认真做些笔记。

谁是策划者？	398
谁来行动？	399
让二者协作	400
做一点文件系统“家务活”	401
记笔记！	402

iv

安装Android SDK和工具

完善环境

要成为测试原生Android应用的主人，必须当心环境。

要把你的计算机变成一个条件宜人的小环境，在这里你可以在虚拟（模拟）设备或真实设备上“放牧”Android应用。为了让你成为合格的牧羊人，管理好这些Android小羊，我们会告诉你如何下载Android软件开发包（Android Software Development Kit，SDK），如何安装一些平台工具，如何创建一些虚拟设备，以及如何安装和卸载应用。

下载Android SDK	404
找到最合适的工具	405
创建一个新的虚拟设备	408
找到正确的PATH	413

索引	417

1 移动Web入门

响应式Web设计



嘿！准备好了吗？要进入移动世界吗？

移动Web开发绝对是一种让人惊心动魄的生活方式。它有十足的魔力，让人兴奋不已，欲罢不能，你会一次次地感受到顿悟的快乐！不过，这里也有谜团，也有困惑。移动技术正在飞速发展，需要了解的东西实在太多了！快抓紧，就要开始我们的移动之旅了，我们会带着你了解一种建设网站的方法，称作响应式Web设计（Responsive Web Design，RWD）。采用这种方法，可以利用你已经掌握的Web技能适当地调整网站，让它们在各种移动设备上同样有非凡表现。

登上“移动”花车

你很可能已经有一部移动手机。我们知道，这不光是因为你买了这本书（顺便夸一句，买得好），还因为现在真的很难找到一个居然没有移动手机的人。

不论你身处何地，都能见到移动手机的身影，比如尼日利亚的农民会用移动手机查看他们的农产品在哪个市场卖价最高，再比如日本的十大最畅销小说就是在移动手机上销售和写作的（没错，作者们确实在用移动手机写小说），可以说，移动手机的使用已经遍及全世界各个角落。

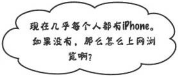
2011年初，全球69亿人使用的移动手机数达到52亿部。使用移动手机的人甚至比用马桶和牙刷的人都多。

就是现在

没错，移动的覆盖面很广，不过并不只是现在才这样，多年来这一直是个庞大的群体，为什么要现在登上“移动”这驾花车呢？

这是因为，iPhone改变了一切。听上去有些夸张，但确实如此。在iPhone出现之前，也有应用商店（App Stores），也有触摸屏和Web浏览器，但是Apple公司首次用一种便于人们理解和使用的方式把它们结合在了一起。







iPhone非常棒，不过出于不同的原因，人们还在使用各种不同的手机。最流行的手机不会一成不变。

我无从得知你读这本书时最火的手机会是什么。3年前，Android还只是部署在雷达上，而在2011年，它已经摇身一变成为全世界最热门的智能手机平台。

移动技术的变化很快，不过有几个方面我们很有信心：

1. **所有新手机上都会有Web浏览器。**

   你可能发现一个新手机里没有Web浏览器，不过你得再仔细看看。现在即使最基本的手机也会内置一些不错的浏览器。所有人都希望能通过手机上网浏览。

2. **移动Web应用会超越桌面Web应用。**

   通过移动手机上网的人数很快就会超越利用计算机上网的人数。实际上，很多人已经发现他们用手机比用PC更频繁。

3. **Web是唯一真正的跨平台技术。**

   iPhone、Android、BlackBerry、Windows Phone、WebOS、Symbian、Bada等，还有很多手机平台，我们无法一一掌握。每个平台都有相应的编程接口，这意味着如果你希望为每一个平台编写软件，就必须每一次都从头开始。

   移动Web当然也存在自己的难题，不过它允许你创建适用于所有平台的内容和应用，在这方面没有哪种技术能够与之匹敌。

所以，事不宜迟，箭在弦上，移动Web就要起飞了，出发吧！

酒吧发生了怪事

Mike经营着一个酒吧，名字很可爱，叫作非凡海象（Splendid Walrus），在当地有一群忠实的粉丝。Mike经常会提供一些特别的啤酒供大家享用，还会在他的网站上特别标出。

在Mike实现他成为酒吧老板的人生梦想之前，曾做过Web开发人员。所以，为非凡海象酒吧建立一个美观大方的网站，这对他来说不算难事。



如果移动手机Web浏览器这么棒……

这个非凡海象网站是Mike几年前建立的，那时移动浏览还很初级，也很少见。这个网站是基于Firefox、Internet Explorer和Safari之类的桌面浏览器构建的，当然也只是在这些浏览器中做过测试。

很多更新的移动浏览器口碑不错。它们越来越精巧，也越来越强大，俨然与那些桌面浏览器相差无几。

……是不是没问题了？

Mike在他的iPhone 4上浏览非凡海象网站，结果让他大跌眼镜。在一个朋友的Android设备上网站的表现也差强人意。



移动Web为什么有这么大差异？



1. **现在有860亿个不同的移动Web浏览器。**

   OK，可能没有那么多。不过，在做移动Web开发时，有时就会有这种感觉。我们只有为数不多的几个领先的桌面浏览器，但移动浏览器就不同了，可能有成百上千个之多。千真万确！

   你以为自己已经完全掌握所有移动浏览器，可能突然之间又会出现一个新的浏览器，比如Thailand。

2. **对Web技术的支持千差万别。**

   在较早的移动浏览器上（或者甚至在能力较逊的设备上运行最新的浏览器时），可能要完全忘掉原本可以信赖的CSS或JavaScript。甚至最新的浏览器也未提供某些支持，或者支持的方式截然不同，也可能存在奇怪的bug。这完全是一个纷乱的蛮荒世界，真的！

3. **移动设备更小，更慢。**

   没错，这一点我们很清楚。较新的移动设备是一些拥有最新技术的便携式计算机。不过与桌面计算机（或笔记本电脑）相比，在处理能力方面它们还是稍逊一筹。移动网络可能很脆弱，而且可能速度很慢，数据传输不一定是免费或不受限的。这说明，要从性能角度考虑漂亮但庞大、富媒体的复杂网站。

4. **移动接口要求重新考虑我们的网站。**

   移动浏览器确实可以呈现桌面网站（只是可能会稍稍有些停顿），但并不能因为能够这么做就应该这么做。现在屏幕变小了，交互和我们的期望也有所不同。

   有移动设备的人可能使用各种各样的输入设备：手指、写字笔、黑莓手机上的小按钮等。不论是写字还是填写表单都很乏味，更何况还可能出现错误。要让用户眯着眼睛在一个原本为桌面浏览器窗口设计的表单中写入内容，这会让他们头疼甚至抓狂。这种痛苦你应该懂的。

Exercise

下面是在Mike的iPhone 4上呈现的非凡海象网站。看起来可不怎么样。你能找出哪些地方有问题吗？标出你发现的问题。



1 ....

2 ....

3 ....

4 ....

Exercise Solution

你发现这些问题了吗？



1. 导航链接都太小，根本无法阅读，无法点击。
2. 嵌入的YouTube视频没有播放。
3. 在这个屏幕上三栏布局感觉有些挤，文本不便于阅读。
4. 屏幕右边界有一个奇怪的空隙。





Frank：等一下。我们知道，Mike用简洁、明确的HTML标记搭了一个不错的架子，还使用CSS尽可能完善地控制了布局和样式。

Jim：那又怎样？这当然很好，很专业，不过对我们改进网站有什么帮助呢？

Frank：嗯，我们先好好想一想。查看他在非凡海象网站中使用的CSS时，我注意到有很多宽度和大小定义，以适应960像素的方框。看起来他是在一个960像素的窗口上设计的这个网站，而且分为三栏。

Jim：……不过大多数移动设备的显示屏都远远小于960像素。另外，对一个比较小的屏幕来说，3栏显得有些太多了。

Frank：所以说……我在想……能不能对移动设备使用不同的CSS？比方说，为适应320像素设计的CSS（这是很多智能手机屏幕的宽度）？另外，是不是可以减少栏数？

Jim：好主意，Frank。不过除了做大量服务器端编程，我想不出有什么办法能够做到。我的意思是说，我们怎么让移动设备使用完全不同的CSS呢？

Frank：知道吗？Jill刚开完Awesome Cool Mobile Web Camp会议回来，还在为那种叫作响应式Web设计（Responsive Web Design）的技术激动呢！

Jim：我怎么能忘呢？她一天到晚都在谈论这个。

Frank：对啊，她说这个技术正在受到Web开发人员的关注，听上去好像（至少在某种程度上）是关于针对不同的情况应用不同的CSS，而不需要做繁重的编程工作。很显然，这对于开发移动网站尤其有用。我记不得具体的细节了，不过我们可以查查看。

响应式Web设计

响应式Web设计（Responsive Web Design，RWD）是Web设计人员Ethan Marcotte积极倡导的一组技术。采用这种方法设计的网站可以根据用户浏览器的环境调整布局，大多会利用CSS做一些精妙的处理。

可以在该Ethan之前为 A List Apart写的一篇关于RWD的文章（http://bit.ly/nRePnj）。

根据某些浏览器条件的当前值，如窗口大小、设备方向或长宽比等，我们可以在不同场合下应用不同的CSS。通过重新考虑设置页面布局的方式，我们可以调整原先那种“以不变应万变”的行列网格布局，以适应各种不同的浏览器窗口大小，使之能更自然地呈现。

要让一个网站漂亮地适应多种不同设备，RWD是最简单、最快捷的方法之一。采用这种技术，你完全可以使用之前已经拥有的一些Web技能。

响应式Web设计主要组成

要构建一个响应式网站，主要包括三大技术：

1. **CSS3媒体查询。**

   计算当前浏览器环境的某些方面，来确定应用哪一个CSS。

   我们可以根据某些方面应用不同的CSS规则，比如浏览器窗口宽度、长宽比和方向等。

2. **流式网格布局。**

   对页面布局元素使用相对CSS比例而不是绝对大小。

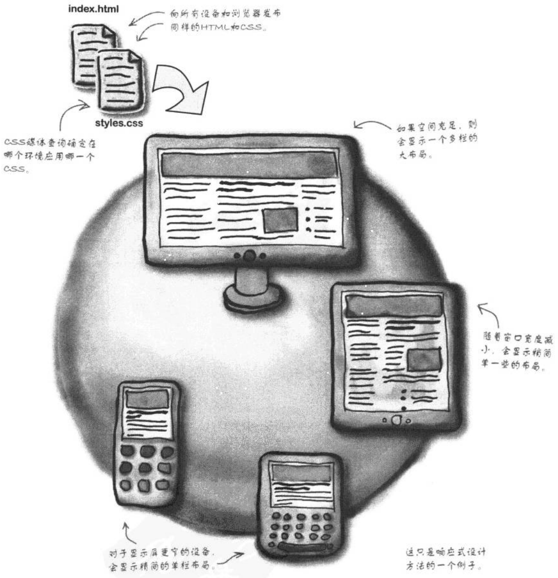
   RWD对列和其他布局元素使用百分比，而不是使用像素单位指定绝对大小。

3. **流式图像和媒体。**

   通过使用一些CSS技巧，使图像和媒体比例适应其容器的大小约束。

   流式图像和媒体会限制在其父元素的范围之内，相应于布局的其他部分按比例缩放。

一个响应式网站例子


index.html
向所有设备和浏览器发布同样的HTML和CSS。
styles.css
CSS媒体查询确定在哪个环境应用哪一个CSS。
如果空间充足，则会显示一个多栏的大布局。
随着窗口宽度减小，会显示精简单一些的布局。
对于显示屏更窄的设备，会显示精简的单栏布局。
这只是响应式设计方法的一个例子。

不同场合使用不同的CSS

如果你做过一段时间的Web开发（而且熟悉CSS），则可能对CSS媒体类型已经并不陌生。我们可以使用@media规则有选择地应用CSS。

CSS文件中的CSS媒体类型声明如下所示：

```
@media screen { /* CSS Rules for screens! */ }
```

"screen"是一个媒体类型。

只有在屏幕上呈现内容时才应用大括号之间的规则。

使用媒体类型有选择地应用CSS还有一种方法，可以在HTML文档的<link>中设置。

```
<link rel="stylesheet" type="text/css" href="print.css" media="print" />
```

只有在一个打印设备（也就是一台打印机）上呈现内容时才应用这个外部样式表中的规则。

"print"是另外一种媒体类型。

像这样引用print媒体类型是创建打印样式表的一种常用方法，打印样式表就是仅在打印内容时才应用的CSS样式。

媒体类型特写

常用（而且有用的）媒体类型包括screen、print和all。另外还有一些不太常用的媒体类型，比如aural、braille和tv。

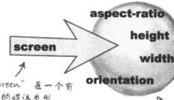
是不是有些好奇？如果你是那种热衷于读技术规范、喜欢满足好奇心的人，则可以看看W3C网站上CSS2中定义的所有媒体类型，网址是www.w3.org/TR/CSS2/media.html。

媒体类型，认识一些媒体特性

你自己有一些特性，比如你的年龄、身高等，媒体类型也一样有自己的特性。非凡海象网站希望建立这样一个规则：要求不得向21岁以下的顾客供应烈酒；或者我们可能想定义一个CSS，只有当浏览器窗口宽度在某个范围内时才应用这个CSS。

很幸运！CSS3为所有常用媒体类型都定义了width、color和orientation媒体特性。所以，再重申一次，媒体类型都有媒体特性。

媒体特性本身用处并不大。我们需要以某种方式向浏览器询问我们关心的某些媒体特性的状态，并相应地做一些处理。这就引入了CSS3媒体查询。



"screen"是一个有用的媒体类型。

"screen"媒体类型的一些媒体特性。

只是，还有很多其他媒体特性。不过，对我们来说，这些是最常用的特性。

CSS媒体查询

"screen"媒体类型，又见面了！

我们要计算"screen"媒体类型的"width"媒体特性。

只有当媒体查询计算为TRUE时才会应用这些CSS规则。

```
@media screen and (min-width:480px) { /* CSS Rules */ }
```

"min-"是一个媒体查询前缀。很直观，它表示我们想查询一个最小宽度。

毫无疑问，还有一个"max-"前缀。

这表示：当前是否在一个屏幕上呈现内容？而且当前窗口宽度是否至少为480像素？

确认？那好！应用这些CSS规则。

再举一个例子：

```
@media print, screen and (monochrome) { }
```

逻辑"或"用逗号表示。没错，这确实让人有些迷惑。

"monochrome"是"screen"媒体类型的一个媒体特性，可以为TRUE或FALSE。

在一个打印机上呈现吗？或者是在一个单色（黑白）屏幕上呈现吗？

确认？那就使用这些样式！

CSS3媒体查询是一些逻辑表达式，用于计算用户浏览器中媒体特性的当前值。如果媒体查询表达式计算为TRUE，就会应用所包含的CSS。

Exercise

解释CSS媒体查询：你可以自己试试！将下面的媒体查询与相应的含义连线。

```
@media all and (orientation: landscape) {}
```

```
<link rel="stylesheet" type="text/css"
href="my.css" media="screen and (color)" />
```

```
@media print and (monochrome) {}
```

```
@media screen and (color) { }
```

对彩色屏幕应用这个外部样式表中的规则。

对黑白打印机应用这些样式。

对彩色屏幕应用这些规则。

对于所有媒体类型，如果是水平方向，则应用这些样式。



你能解释这些媒体查询吗？

你是不是不清楚"all"媒体类型？没错，如果我们想查看所有媒体类型的相同媒体特性，就会用到"all"。

```
@media all and (orientation: landscape) {}
```

```
<link rel="stylesheet" type="text/css"
href="my.css" media="screen and (color)" />
```

```
@media print and (monochrome) {}
```

```
@media screen and (color) { }
```

对彩色屏幕应用这个外部样式表中的规则。

对黑白打印机应用这些样式。

对彩色屏幕应用这些规则。

对于所有媒体类型，如果是水平方向，则应用这些样式。

OK。现在我了解媒体查询了，甚至可以写我自己的媒体查询。不过接下来我要做什么？怎么为移动设备写CSS呢？



CSS:多不同才算不同？

我们已经有了一个工具，可以对不同情况应用不同的CSS。不过，现在该做什么呢？

别着急。我们确实要写一些对移动设备友好的CSS，不过不用一切从头开始。另外也不需要为移动设备分别编写完全不同的CSS，我们可以分享已有的很多CSS。

要生成对移动设备友好的布局，我们要做到：

- [ ] 检查splendidwalrus.com的当前布局，分析它的结构。
- [ ] 找出需要修改的布局部分，从而在移动浏览器上能有更好的表现。
- [ ] 为找出的这些元素生成适应移动设备的CSS。 ← 只有当布局元素针对移动设备需要有所不同时，我们才会为这些布局元素编写不同的CSS。
- [ ] 组织CSS，使用媒体查询选择性地应用移动和桌面CSS。

当前非凡海象网站的结构

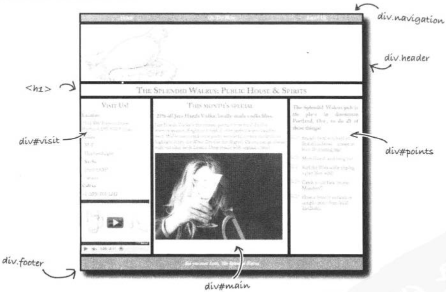
可以大致看看chapter1目录中非凡海象网站的index.html文件。发挥你的想象力，去掉具体的内容，可以看到一个基本HTML页面结构，如下所示：

```
<div class="navigation">…</div>
<div class="header">…</div>
<h1>…</h1>
<div id="visit" class="column">…</div>
<div id="points" class="column">…</div>
<div id="main" class="column">…</div>
<div class="footer">…</div>
```

3个主要内容栏

在标记中右栏放在中间栏前面，这是Web设计界常用的一个技巧，这样更容易使用CSS floats处理布局。

目前桌面CSS定义了以下页面布局：



下面来看看定义这个布局的CSS，确定哪些方面需要修改以适应移动设备的屏幕大小。

分析当前CSS

打开非凡海象网站的styles.css文件。

这个文件最前面有一大堆CSS，不过先不用担心这些。桌面和移动浏览器可以共享相同的颜色、排版和样式。

我们关心的是有关结构的CSS，它们在靠近文件最下面的位置。

我们只对CSS文件中有关结构的部分感兴趣。

```
/* Structure */
body, .header, .navigation, .footer {
  width: 960px;
}
.header, .navigation, .footer {
  clear: both;
}
.column {
  margin: 10px 10px 0 0;
}
.navigation {
  min-height: 25px;
}
.navigation ul li {
  width: 320px; /* 960/3 */
}
.header {
  background:url(images/w.png) no-repeat;
  height: 200px;
}
#visit {
  width: 240px;
  float: left;
}
#points {
  width: 240px;
  float: right;
}
#main {
  margin: 10px 260px 0 250px;
  width: 460px;
}
```

主体（body）宽度为960像素。页眉、页脚和导航元素占满整个主体宽度。

因为这些元素占满了整个主体宽度，所以一定要确保它们旁边没有其他元素。

clear:both的作用就是确保这些元素从新的“一行”开始，也就是说，旁边没有其他元素。

各栏（visit、right和points）上面有10px的边距，右边也有10px的边距（即“gutter”，栏内文字与边缘的间隔）。

导航链接放在一个<ul>中。将它水平放置，并让每个<li>占页面宽度的1/3。

每个<li>占页面宽度的1/3，因为这里有3个链接。

页眉有一个背景图像，所以需要保证200px高，以便完全显示这个图像。

左栏和右栏分别为240像素宽，并定义为浮动。

看起来主栏宽度应该是480像素（960像素减去分别为240像素宽的左栏和右栏）。不过，考虑到各栏之间的两个10像素的间隔，所以实际上主栏宽度为460。

主栏利用边距来定位，它不是浮动的。

左边距为260像素，右边距为250像素，这就指定了它在窗口中的位置。

哪些需要改变？

1 让页面和页面结构元素能放在320像素以内。

正像Frank在第9页提到的，320像素是一个常用的移动设备屏幕分辨率。

2 将三栏减为单栏。

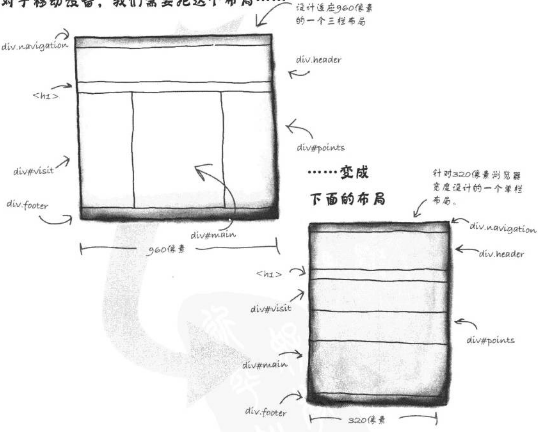
原来桌面布局中的三栏在移动设备屏幕上看起来“很挤”。

Relax

我们可不希望重写所有CSS。

只需要调整其中一些结构布局元素。其余的（包括排版、颜色和诸如此类的内容）基本上都可以保持不变。

对于移动设备，我们需要把这个布局……



……变成下面的布局

找出需要修改的CSS

要为我们的移动版本调整以下突出显示的代码。

```
/* Structure */
body, .header, .navigation, .footer {
  width: 960px;
}
.header, .navigation, .footer {
  clear: both;
}
.column {
  margin: 10px 10px 0 0;
}
.navigation {
  min-height: 25px;
}
.navigation ul li {
  width: 320px; /* 960/3 */
}
.header {
  background:url(images/w.png) no-repeat;
  height: 200px;
}
#visit {
  width: 240px;
  float: left;
}
#points {
  width: 240px;
  float: right;
}
#main {
  margin: 10px 260px 0 250px;
  width: 460px;
}
```

styles.css

要改变页面以及页眉、导航和页脚元素的宽度。

移动版本中不需要这个规则（不过这也不会有什么影响）。

由于在移动布局中没有浮动的元素，所以不需要clear。

移动布局中的"columns"会垂直放置，而不是水平放置。下面在各栏之间（垂直）增加一些空间，去除原来栏文字与边缘的间隔。

这实际上没什么问题，我们希望导航链接至少有这个高度。

需要调整导航链接的宽度，以适应更小的屏幕。

页眉要使用相同的背景图像，所以这可以保持不变。

看起来可能需要调整这里的200像素高度，不过实际上并不需要，因为我们会使用同样的图像。

需要删除这些float设置，并改变visit和points栏的宽度。

不再需要边距来定位（#main会占满整个宽度），不过需要改变宽度。

创建移动CSS的步骤

1. 修改突出显示的CSS规则的宽度。
2. 删除我们不需要的CSS规则。
3. 抽取出公共CSS规则。 ← 稍后会做这个工作。



移动CSS磁贴

使用磁贴建立移动CSS。

```
body, .header, .footer, .navigation {

}
.column {

  border-bottom: 1px dashed #7b96bc;
}
.navigation ul li {

}
#visit, #points, #main {

}
```

↑ 在页面上放置各栏时（现在是垂直放置），这个边框可以增加一些视觉分离效果。

```
width: 320px;
width: 106.6667px;
margin: 10px 0;
width:320px;
```



移动CSS磁贴答案

```
body, .header, .footer, .navigation {
  width: 320px;
}
.column {
  margin: 10px 0;
  border-bottom: 1px dashed #7b96bc;
}
.navigation ul li {
  width: 106.6667px;
}
#visit, #points, #main {
  width:320px;
}
```

完工了！移动CSS

大功告成！这4个CSS规则正是我们需要的移动布局。现在需要确保移动设备使用这些规则。

怎么做到呢？我们的老朋友媒体查询可以帮忙！稍后我们会生成一个媒体查询，将这个CSS应用到浏览器窗口小于等于480像素的设备。

为什么是480像素？这是很多流行的智能手机的“长边”（即“水平方向”）的分辨率。



它们并没有消失……

……只是不再需要把它们包含在我们的移动CSS中。为什么呢？因为对于这两个布局（桌面布局和移动布局），这些CSS是一样的。

我们将把这些公共的CSS放在媒体查询之外，这样就不必在两个不同的地方放置相同的CSS规则。现在就来做这个工作。

其他结构CSS

共享的结构CSS

看到了吧？我说的没错吧！实际上CSS根本没有消失。这里就是我们在18页上看到的结构CSS，只是被抽出来了。

```
.header, .footer, .navigation {
  clear: both;
}
.header {
  background:url(images/w.png) no-repeat;
  height: 200px;
}
.navigation {
  min-height: 25px;
}
```

桌面结构CSS

还需要为桌面浏览器建立漂亮的CSS！

去掉那些公共的结构CSS规则之后，这里可以看到专门针对**桌面浏览器的CSS结构**。

我们要使用一个媒体查询，使得只有宽度在481像素以上的视窗才适用这个CSS。

```
body, .header, .footer, .navigation {
  width: 960px;
}
.column {
  margin: 10px 10px 0 0;
}
.navigation ul li {
  width: 320px; /* 960/3 */
}
#visit {
  width: 240px;
  float: left;
}
#points {
  width: 240px;
  float: right;
}
#main {
  margin: 10px 260px 0 250px;
}
```

下一步做什么？

为了创建同时适用桌面和移动浏览器的结构CSS，下面来看为此建立的任务列表：

- [x] 检查splendidwalrus.com的当前布局，分析它的结构。
- [x] 找出需要修改的布局部分，从而在移动浏览器上能有更好的表现。
- [x] 为找出的这些元素生成适应移动设备的CSS。
- [ ] **组织CSS，使用媒体查询选择性地应用移动和桌面CSS。**

集成在一起

我们通过以下方式集成这个更新版本的styles.css。

```
颜色、排版、基本布局

公共结构CSS（第21页）

@media screen and (min-width:481px) {

桌面结构CSS（第21页）

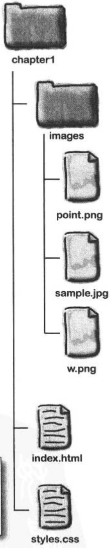
}
@media screen and (max-width:480px) {

移动结构CSS（第20页）

}
```

对这部分CSS我们没有做任何改动。

styles.css



最后一点……

index.html文件中需要一个视窗<meta>标记。这些标记可以告诉浏览器如何“放大缩小”来呈现内容。后面我们谈到这些标记，不过现在只要记住一点：肯定会需要下面这个标记。

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<title>The Splendid Walrus: Public House and Spirits</title>
```

index.html

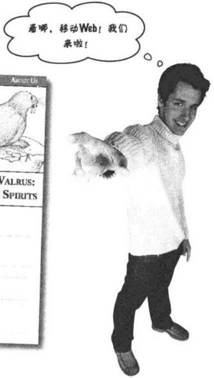
动手做！

1. **编辑index.html文件。**
   加入第22页的<meta>标记。
2. **打开styles.css文件。**
   要替换文件末尾附近的结构CSS规则。删除结构元素的现有规则。
3. **增加公共规则。**
   增加第21页上的共享结构CSS规则。
4. **增加面向桌面和移动浏览器的CSS。**
   增加桌面规则（第21页）和移动规则（第20页）。
5. **把面向桌面和移动浏览器的CSS放在媒体查询中。**
   增加媒体查询（第22页）。

试一试

一旦完成上述修改，就在你的桌面浏览器中加载index.html文件，调整窗口大小，使宽度小于481像素，可以看到适用移动设备的布局。





在桌面浏览器中页面看上去仍然是一样的……

……直到浏览器窗口宽度变到小于481像素时。现在可以看到适用于移动浏览器的布局了！

这是个不错的开端，不过还存在一些问题。如果向下滚动iPhone，就会看到照片相对于页面来说太大了。这会破坏布局。



Frank：真让人头疼。我还以为我们创建的移动CSS能解决这个问题呢。

Jill：你的CSS能建立适用于较小屏幕的布局，不过还是太严格。我的意思是说，如果我把iPhone水平放置，那么看看会发生什么。

Frank：哎呀，布局宽度还是320像素……不过这可是480像素宽的屏幕呀。难道说对每一种可能的视窗大小我都得编写不同的CSS吗？

Jill：哈，这里响应式Web设计就有用武之地了。目前，你为当前窗口宽度在480像素以下的所有浏览器提供了一个大小严格限定的结构，而不论这个特定浏览器窗口的具体宽度是多少。这样很不灵活，我是说，并不是所有移动设备都有320像素宽的浏览器窗口。

响应式设计可以帮助我们根据不同的情况调整布局。并不是限定元素的绝对大小。也就是说，就像现在这样在结构CSS中使用基于像素的度量方式指定大小吗，完全可以使用一个比例布局，这样可以更好地适应不同的用户。

Frank：比例？是不是就像CSS中的em和百分比之类的东西？

Jill：对，差不多吧。建立布局时我们可以使用百分比而不是像素。这样一来，内容就可以伸展和收缩来填充可用的空间—有些像灌进空隙里的水。正因如此，这种布局通常被称为流式布局（fluid layout）。另外，顺便说一句，这也能帮助我们调整那个“出格”的图像。

Frank：这么说，我们用媒体查询做的所有工作都是白费工夫了。

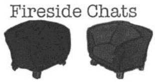
Jill：那倒不是！媒体查询可是实现响应式设计的重头戏。

这个照片宽度大于320像素，超出了布局。



YouTube视频在这个iPhone上还是不能播放。

采用响应式设计，可以在更多设备和浏览器上提供让人感觉更舒服的应用。要转向响应式设计，下一步是把之前基于像素的固定布局转换为一种成比例的流式网格布局。



今晚话题：当固定网格布局遇到流式网格布局

流式网格布局：

固定网格布局：

嘿，固定网格布局。我知道你很有阅历，可谓见多识广。不过，绝无冒犯之意，你的处世哲学确实还有些问题。

什么问题？像你这样初出茅庐的小子也懂哲学？

我是说……你有点老气。你总是严格定义尺寸大小，这让你看起来很脆弱，也很不灵活。也就是说，你不是响应式的。

响应式？

你不能对用户环境中的改变做出响应。你总是老样子。

传统有什么错？我就是要提供设计者期望的效果！960像素宽，另外左右栏宽度分别240像素。

你的用户可就倒霉了。对于不同的浏览器窗口大小，看看你的表现吧，不是少了内容，就是留出大块空白。

如果用户不满意，他们应该自行改正，调整一下，改用一个标准的浏览器才对。

在你那个时代这也许是个办法，不过现如今浏览器和设备种类实在太多了。如果你愿意放弃这种严格限定像素的布局（嘿，毕竟它们是古老的印刷年代流传下来的），而让你的内容像水一样流入浏览器窗口中的可用空间，你就能适用各种各样不同的情况。

好吧，小家伙，让我看看你能做什么。

固定宽度布局到底有什么问题？

960像素窗口中显示的网站。



如果这个世界的浏览器窗口都一样大，这将是一个安全、美丽的世界，设计者们完全可以通过限定像素控制网站的外观。

遗憾的是，Web从来没有这么容易控制。有时我们想要根据一些“标准的”窗口宽度（比如640、960或1024像素）来设计。不过大多数情况下这都只是一个幻想：根本没有标准的浏览器窗口大小。这还只是桌面浏览器的情况，要是考虑移动设备，情况就更复杂了。

当然，非凡海象网站的固定布局在960像素宽的窗口中看起来还是不错的。

这个布局不能适应其他窗口大小

不过在一个较窄的窗口中，看看会发生什么。栏宽还是一样，这说明有些内容会被切掉，用户不得不水平滚动屏幕。真糟糕。

在700像素宽的窗口中看到的网站。



如果不滚动，右栏就看不到。

左右栏……总是240像素宽。

在较宽的窗口中，整个布局仍然只有960像素宽，这就会浪费屏幕右侧的大块空白区域。嗯，真不怎么样。

整个宽度限制为960像素。



在1200像素宽的窗口中看到的网站。

为什么流式网格布局更好？

流式网格布局使用比例单位（百分数）而不是像素来表示宽度。我们仍然能提供设计者预想的效果，让左右栏分别占页面的1/4，只需要定义它们的宽度为25%，而不是240像素。



在960像素宽的窗口中查看流式网格布局。目前看起来没有什么不同，是不是？

随着窗口改变大小，布局会相应调整

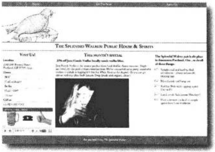
对于不同的窗口宽度，内容会像水一样流入，填充布局中的可用空间。左栏和右栏总是占据窗口的25%，在较窄的窗口中不会剪裁内容，在较宽的窗口中也不会有空白区域。



700像素宽的窗口中查看流式网格布局。

左栏和右栏按比例指定：占窗口宽度的25%。

1200像素宽的窗口中查看流式网格布局。



BRAIN BARBELL

我们要把桌面和移动CSS从固定网格布局转换为流式网格布局。想想看这对于解决Jill在移动浏览器上查看网站时发现的问题有哪些帮助？

随大流

要解决Jill发现的问题，转向响应式设计，还需要做很多工作。

- ☐ 使用比例宽度而不是固定宽度，将基于像素的布局转换为流式网格布局。
- ☐ 设置默认字体大小为100%，这样页面的字体就能按比例缩放。 ← 如果要建立流式网格布局，肯定希望字体也是灵活的。
- ☐ 修正无法播放的YouTube视频。
- ☐ 修正过宽的图像。

流公式

要把一个基于像素的布局转换为按比例的流式网格布局，可以使用以下公式：

$$\frac{\text{目标}}{\text{上下文}} = \text{结果}$$

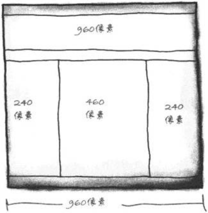
目标：元素的大小，用像素单位度量。

上下文：包含元素的"上下文"的大小，用像素单位度量。

结果：新的比例CSS规则（一个百分数）。

详细分析

下面借用非凡海象网站的桌面布局来分析这是什么意思。



首先来看作为比例计算基础的上下文。在这里，我们的参考设计为960像素宽。我们希望得到的流式网格布局与当前设计比例相同。所以要根据这个960像素基准来计算。

导航、页眉和页脚都横跨整个页面宽度，因此可以很容易地应用流公式计算！

$$\frac{\text{960像素}}{\text{960像素}} = 100\%$$

继续流式转换

相对于包含上下文的960像素，左栏和右栏宽度都为240像素。要得到比例度量，可以再次使用流公式：

$$\frac{240\text{像素}}{960\text{像素}} = 25\%$$

左右栏各占页面宽度的1/4。所以这很直观，不是吗？

中间的主栏稍有不同。它不是浮动的。相反，要利用页边距为元素定位。不过这也没关系。我们仍然可以使用流公式将基于像素的页边距大小转换为百分数：

$$\frac{250\text{像素}}{960\text{像素}} = 26.0416667\%$$

当前的CSS规则。

```
#main {
  margin: 10px 260px 0 250px;
}
```

$$\frac{260\text{像素}}{960\text{像素}} = 27.0833333\%$$

更新后的CSS规则

```
#main {
  margin: 10px 27.0833333% 0 26.0416667%;
}
```

Exercise

将styles.css中有关结构的桌面CSS规则转换为比例宽度。可能需要编辑面向窗口宽度481像素以上的媒体查询中的6个CSS规则。

there are no Dumb Questions

问： 这些数里为什么有这么多小数位？需要这么多位吗？

答： 这里我们主要为了展示纯正的流式网格布局方法，完全按计算器得到的数来建立CSS单位。

实际上，浏览器会对这些过长的数四舍五入。另外，要特别注意，不同的浏览器四舍五入的方式还稍有不同。

所以，这要看你的了。可以考虑把这些数四舍五入为小数点后有一位或两位。另一种方法是在布局中保留这些数，一两个百分数根本不会有太大影响。布局不会太精确，不过，要避免四舍五入问题的一些陷阱，另外计算不要太严格。

问： 等一下。为什么.main的上边距还是10像素？是不是……弄错了？

答： 垂直布局与水平布局完全是两码事。我们不能使用960像素上下文，因为设计的高度通常不会是一个已知量，使用百分数的垂直布局会很麻烦，不同的浏览器中会采用不同的（有时甚至是拙劣的）方式支持。对于垂直边距，像这样使用像素是完全可以的。

Exercise Solution

下面来看这个流式桌面CSS。

```
@media screen and (min-width: 481px) {
  body, .header, .footer, .navigation {
    width: 100%;
  }
  .column {
    margin: 10px 1.04166667% 0 0;
  }
  .navigation ul li {
    width: 33.333333%;
  }
  #visit {
    width: 25%;
    float: left;
  }
  #points {
    width: 25%;
    float: right;
  }
  #main {
    margin: 10px 27.0833333% 0 26.0416667%;
  }
}
```

$\frac{960}{960} = 100\%$

$\frac{10}{960} = 1.04166667\%$

$\frac{320}{960} = 33.333333\cdots\%$

$\frac{240}{960} = 25\%$

这在第29页已经解释过。

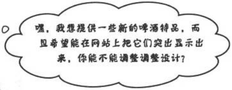


别着急！

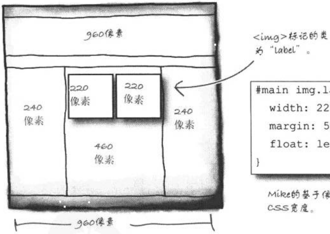
有时“上下文”可能会改变……

上下文切换





Mike提供了一个新的每月特品，他希望在非凡海象网站上发布出来。不过不要像现在这样在主栏中通过文本和一个图像来表现，他想浮动显示两款非常特殊的限量版啤酒的标签，让它们彼此相邻。在基于像素的参考设计中，看上去是这样的：




```
#main img.label {
  width: 220px;
  margin: 5px;
  float: left;
}
```

你是不是认为将这些图像宽度转换为流式宽度的公式应该是这样：

$$\frac{220\text{像素}}{960\text{像素}} = 22.916667\%$$

你觉得这个公式对吗？

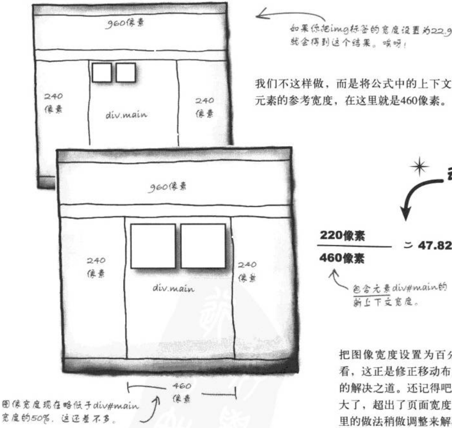
这个图片怎么了？

上下文已经改变了！



如果把图像宽度设置为22.9166667%，它们会占据22.916667%，不过是其包含元素的22.916667%。

这些图像的包含元素不再是主体（宽度100%，或参考设计中的960像素），现在的包含元素是div#main，它的宽度大约是460像素（按比例来讲就是960像素的47.91667%）。所以刚才我们指定的图像宽度只是460像素的23%还略少一点，这太小了！



我们不这样做，而是将公式中的上下文设置为包含元素的参考宽度，在这里就是460像素。

动手做！

$$\frac{220\text{像素}}{460\text{像素}} = 47.826087\%$$

包含元素div#main的新上下文宽度。

把图像宽度设置为百分数？仔细想想看，这正是修正移动布局另外一个问题的解决之道。还记得吧？有一个照片太大了，超出了页面宽度。我们可以把这里的做法稍做调整来解决这个问题！

流式图像 ← 和媒体！

别看这个CSS很简短，能量可不小：

```
img, object {
  max-width: 100%;
}
```

只是做了这样一点小小的补充，就能避免图像或嵌入的媒体对象超出其包含元素。因为它们限制为100%宽度，也就是其包含元素宽度的100%，图像和媒体会听从其父元素的安排，不会试图超出边界。太棒了！

流式图像和媒体像流式网格布局一样，会在布局内按比例缩放。

悲伤的告别……

大多数情况下，不做出点牺牲就不会有收获。要想在图像和媒体上使用流技术，我们必须放弃我们的老朋友：width和height属性。

上面的CSS规则会覆盖width属性，不过不会影响height属性。这说明，如果我们使用了height属性和width属性，最后就会得到一个宽度缩放而高度不变的图像。最终结果是一个怪异的挤压变形的图像（长宽比有误）。

没错，对于这个问题也有另外一些解决方法，删除这些属性并不是上策。不过我们现在打算先放弃height属性和width属性。

Watch it!

流式图像不是一种“万全的”技术。

可以在较窄的屏幕上压缩一个图像，不过这并不意味着从其本质来讲它有什么变化。一个800KB JPEG的图像仍是800KB JPEG，即使把它压缩到一个120像素宽的栏中，它仍是那么大。

在第2章中，我们将讨论为不同设备和浏览器传送不同图像的技术，以节省本来可能会浪费的带宽和处理器能力（需要处理器完成具体的缩放）。

不管怎样，这仍然是一个强大的技术，绝对值得你收入囊中。

成功了吗？

我们在朝着适应更多设备的响应式设计前进。不过，还有一些问题需要解决：

- [ ] 使用比例宽度而不是固定宽度，将基于像素的布局转换为流式网格布局。
- [ ] 设置默认字体大小为100%，这样页面的字体就能按比例缩放。
- [ ] 修正无法播放的YouTube视频。
- [x] 修正过宽的图像。

还需要转换移动CSS（之前已经转换了桌面CSS）。

这是使用流式图像技术做到的。

转换移动CSS

还有一些移动CSS规则需要转换为流式规则。

```
@media screen and (max-width: 4
  body, .header, .footer, .navi
   {
     width: 320px;
   }
.column {
    margin: 1em 0;
    border-bottom: 1px dashed #7
 }
.navigation ul li {
   width: 106.6667px;
 }
 #visit, #points, #main {
   width: 320px;
 }
```

当前（固定）

```
@media screen and (max-width: 480px) {
  body, .header, .footer, .navigation {
    width: 100%;
  }
.column {
    margin: 1em 0;
    border-bottom: 1px dashed #7b96bc;
 }
.navigation ul li {
   width: 33.333333%;
 }
 #visit, #points, #main {
   width: 100%;
 }
```

更新后（流式）

嘿！这里是比例规则，这两个规则与桌面CSS规则完全相同（见第30页）。也许可以把它们放在公共结构CSS中，而不是分别放在两个不同的地方。

细节，细节

要让更新版本的非凡海象网站充分实现响应式设计，下面来考虑剩下的几个细节问题。

建立灵活的字体

到目前为止，我们的布局已经可以适应性调整，不过字体还是很古板，很固定。就像阴阳对应一样，百分数是流式宽度（作为“阴”），对应像素固定宽度（作为“阳”），em是按比例指定字体大小的单位。Mike在原来的CSS中已经使用了em，所以我们只需要向<body>元素增加以下规则来保证彻底全面：

```
body {
  background: #f9f3e9;
  color: #594846;
  font: 100% "Adobe Caslon Pro",
  "Georgia", "Times New Roman", serif;
}
```

这个基准字体大小重置就像是画龙点睛，它建立了一个明确的参考，CSS中的其他字体大小都依此来定义。这不算是刻不容缓的大问题，只是为了保证整齐！

字体大小特写

通过对<body>元素CSS规则的编辑，我们将页面的基准字体大小设置为100%。不过100%到底是什么意思？下面给出一个简便（只是近似）的经验式：

1em = 100% ≈ 12pt ≈ 16px

不过要记住我们的目标是让内容适应用户的环境。如果用户改变了浏览器的字体大小，100%就会表示为一个完全不同的绝对大小。

另外要记住，移动设备上的字体很复杂，有些情况下，1em可能等于一个（明显）不同的点或像素大小。

修正YouTube视频

很多移动设备不支持Adobe Flash。嵌入式YouTube的标记已经过时了，YouTube现在提供了一个基于iframe的嵌入代码，在iPhone（和其他现代设备）上也能很好地工作。我们需要编辑index.html文件，替换当前的嵌入代码。

这个技术并不仅限于Flash！其他媒体也可以采用这种方式转换为流式。

```
<object width="230" height="179"
type="application/x-shockwave-flash"
data="http://www.youtube.com/v/O-
jOEAufDQ4?fs=1&amp;hl=en_US&amp;rel=0"><embed
src=… /></object>
```

不要用这个（只面向Flash）的版本。

要用这个代码！

```
<iframe src="http://www.youtube.com/embed/O-
jOEAufDQ4" style="max-width:100%"></iframe>
```

YouTube的新嵌入代码会根据浏览器确定所使用的适当的视频格式。它可以为支持的设备（如Miki的iPhone）提供HTML5视频而不是Flash。只需要从视频页面中的“embed”段抽取出这个更新的代码段。

记住你的责任

流式图像的大小调整实际上不会减少图像的文件大小，类似地，流式媒体也不会改变实际媒体的大小。要由你来决定，相对于文件大小和播放视频所需的处理器能力，在移动设备上使用视频（或其他多媒体）是否值得。



Long Exercise

好了！下面完成所有这些修改，使用响应式Web设计技术，让非凡海象网站成为一个对移动设备友好的网站。

编辑styles.css文件

1. 编辑低分辨率设备媒体查询中的CSS规则。把这些固定宽度转换为比例宽度（第34页）。

2. 由于面向移动设备和桌面浏览器的CSS规则按比例度量，找出二者公共的结构CSS规则（第34页）。将这些规则从媒体查询特定的段中删除，把它们放在CSS文件的公共结构段中。

3. 增加第33页上的CSS，实现流式图像（和媒体）。

4. 更新<body>的CSS规则，增加一个比例字体大小基准（第35页）。

编辑index.html文件

5. 用第35页上更漂亮的iframe版本替换只适用Flash的嵌入YouTube。

前进吧！

6. 保存所做的修改，在任何Web浏览器中加载index.html页面。

试着调整浏览器窗口大小，观察内容如何适应这些变化。

there are no Dumb Questions

问：OK。我已经在CSS见过em作为单位，不过还不太理解。到底什么意思？

答：1em就是当前上下文中当前字体大小的一种表示（这可能不会让人留下太深的印象）。听上去没什么意思。不过，相对于它定义你的字体大小时，魔力就会显现出来。所以，如果设置<h1>元素按1.5em显示，这将是其包含元素的基准字体大小的150%。

实际上可以使用同样的流公式由固定的字体大小生成基于em的流式字体大小。在一个基准字体为16点的上下文中，要创建一个18点字体的流式版本，可以这样计算：18/16 = 1.125em（目标/上下文 = 结果）。

问：等一等。为什么是1.125em而不是112.5%？

答：主要是一直以来都是如此，也是为了更清楚，不过em在跨平台方面确实做得更好。常用的Web实践是按百分数定义块元素宽度，而用em定义字体大小。除了极少的几个例外，对于字体大小来说，百分数和em可以互换。对于那些例外：我们将<body> 元素的字体大小（font-size）设置为100%来涵盖这些例外情况。

问：除了移动Web，CSS媒体查询还有没有其他用途？

答：当然有。举个例子：移动设备通常代表屏幕分辨率的最低端，一些更新的宽屏监视器和电话则有很高的分辨率。有时有必要为这些窗口宽度调整布局（比如说增加更多栏）。

问：到底是什么导致第7页上练习中屏幕右边出现奇怪的空隙？

答：这是iPhone特定的问题。显示一个“放大”的Web页面时（也就是说，让它相当于一个桌面浏览器），iPhone上的移动Safari会假设视窗宽度为980像素。在针对移动设备进行优化之前，非凡海象网站布局宽度是960像素，这就会在右边留出一个难看的20像素宽的空隙。

问：“CSS3媒体查询”中的“3”是什么意思？

答：CSS有很多个版本。你可能以为有3个版本，不过情况更复杂一些。CSS2在1998年作为一个“推荐版本”发布，这是大多数Web开发人员长期以来熟悉的版本。媒体类型就是在CSS2引入的。

CSS3与CSS的早期版本完全是两码事，因为它是模块化的，大约有40个不同的模块，而不是一个庞大的复杂的规范。幸运的是，媒体查询模块是比较完整和稳定的模块之一。

问：既然CSS3还没有最终“完成”，浏览器提供支持吗？

答：就像之前说的，现在我们面对的还是一片蛮荒。人们已经越来越广泛地接受了CSS3的一些比较完整的部分，不过，稍后会看到，这与我们憧憬的移动世界还相距甚远。

问：为什么移动布局中右栏在主栏前面出现？

答：在HTML标记中，div#points内容出现在div#main内容前面。在桌面布局中，使用了floats来对#points 内容定位，让它出现在#main内容的右边。移动布局没有使用floats，因此，这些内容会按HTML中出现的顺序显示。

问：你略过了媒体类型的@import语法。能对媒体查询使用@import语法吗？

答：没有多少人喜欢@import语法。所以，我们谈得很少，实际上，我们甚至没有提到它。不过，你说的没错，对于根据媒体类型包含样式表和对于媒体查询来说，这种语法是完全可用的。

下面来逐步分析得到的CSS文件，看看有哪些改变。

这部分代码放在靠近CSS文件最上面的位置。

```
body {
  background: #f9f3e9;
  color: #594846;
  font: 100% "Adobe Caslon Pro", Georgia,
    "Times New Roman", serif;
}
```

为了节省篇幅，这里不再显示这些规则。

(……共同的排版、颜色、边界等（共同的非结构CSS）……)

共同的结构CSS

```
.header, .footer {
  clear: both;
}
.header {
  background:url(images/w.png) no-
repeat;
  height: 200px;
}
.navigation {
  min-height: 25px;
}
img, object {
  max-width: 100%;
}
.navigation ul li {
  width: 33.333%;
}
.header, .footer, .navigation {
  width: 100%;
}
```

这里是针对不同分辨率的结构CSS。

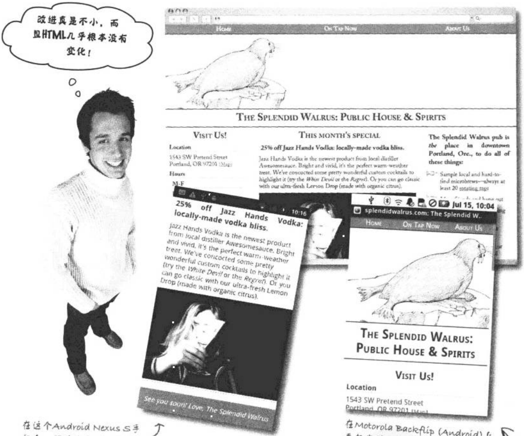
面向较大浏览器窗口的结构CSS（如桌面浏览器）。

```
@media screen and (min-width:481px) {
  .column {
    margin: 10px 1.04166667% 0 0;
  }
  #visit {
    width: 25%;
    float: left;
  }
  #points {
    width: 25%;
    float: right;
  }
  #main {
    margin: 10px 27.0833333% 0 26.0416667%;
  }
}
```

面向较小浏览器窗口的结构CSS（例如，移动设备）。

```
@media screen and (max-width:480px) {
  .column {
    margin: 1em 0;
    border-bottom: 1px dashed
#7b96bc;
  }
  #visit, #points, #main {
    width: 100%;
  }
}
```

得到一个响应式网站！



在这个Android Nexus S手机上，照片大小刚好合适，这要归功于流式图像技术。

在Motorola Backflip (Android)上看起来效果很不错。

BRAIN POWER

在针对移动设备优化的CSS中我们使用了同样的页眉背景图像（尽管实际上它被去掉了）。你能想办法利用RWD技术使用一个不同的图像来节省带宽和提高性能吗？

响应式设计也是一种精神状态



BULLET POINTS

- 移动Web与早先的西部蛮荒地区并没有不同，同样是充满了惊奇和冒险。移动Web浏览器世界包容万象，多姿多彩，有时甚至荒诞不经。
- 我们可以在移动设备上像在一个“传统”浏览器上一样使用同样的布局，不过这并不意味着必须这么做。
- 响应式Web设计（Responsive Web Design，RWD）是一组方法，可以使我们的Web内容适应用户，而不是反过来（强制用户查看严格格式化的页面）。
- RWD是CSS3媒体查询、流式网格布局和流式图像共同构成的一个组合。这也是考虑布局和内容的一种方法。
- CSS3媒体查询允许我们选择性地根据相关媒体特性的当前值对不同的用户环境应用CSS。
- 媒体类型［例如，screen（屏幕）、print（打印设备）、projection（投影）］都有媒体特性（width、color、monochrome、orientation）。我们的媒体查询中就会计算这些媒体特性。
- CSS媒体查询是一个逻辑表达式。计算为TRUE时，就会应用所包围的CSS规则。
- 流式网格布局是指使用比例宽度而不是固定宽度，使得页面的内容可以缩放，自然地流入页面布局，适应各种不同的窗口宽度。
- 流式图像是一种CSS技术，保证当父元素宽度小于图像（或媒体）的宽度时，这个过大的图像（或媒体）不会“超出”其父元素宽度。图像和媒体会随父元素的收缩而收缩。
- 在<body>元素上使用一个简单的font-size重置，按em或百分数定义字体大小，可以保证字体是流式的。

2 真正的响应性

移动优先响应式Web设计

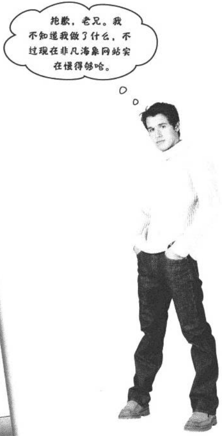


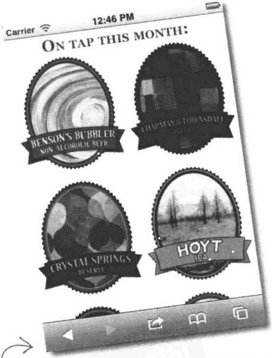
这是一个漂亮的移动网站。不过漂亮只是表面。在内部，它根本不是那样。它可能看上去像是一个移动网站，不过仍然是一个披着移动外衣的桌面网站。如果我们希望这个网站成为移动世界里一道炫目的闪电，就要先从移动开始。首先分解当前的网站，找出隐藏在移动外衣下的桌面成分。我们会做个大扫除，从头开始改头换面，从构建基本内容直到建立一个桌面视图。完成后，你会得到一个不论屏幕大小都可以优化的页面。

你以为该庆祝胜利了……

Mike陷入恐慌。作为一个前Web开发人员，他总想对他的网站做些修补，但一直在克制这种冲动。不过他终于按捺不住，决定稍稍做些修改。他觉得好像破坏了非凡海象网站，现在还需帮助。

Mike为On Tap Now页面上的所有新品增加了图片。除了增加图片外他没有修改任何代码，但是现在移动手机上页面加载得非常慢。实在是太慢了，已经有顾客开始投诉了。





查看On Tap Now页面（http://hf-mw.com/ch2/chapter2/ontap.html。

这算问题吗？我们怎么知道？

Jim：可怜的Mike。会水的才会淹死，他知道得太多了，所以才会遇上麻烦。

Frank：说得是。不过对于现在的情况我不太清楚他哪里做错了。移动手机的网络连接和处理器速度本来就慢一些，页面加载比较慢也是理所当然的啊。

Jim：有道理，不过看起来还是比预期慢太多了。Mike说即使有WiFi，加载速度也无法忍受。而且要知道他有一部非常高级的新款智能手机。

Joe：嗯……听上去我们至少应该看看有没有什么问题明显导致速度减慢。

Frank：我们怎么知道发生了什么呢？可能是网络问题，也可能是手机和服务器之间的很多其他环节出了问题。

Joe：我一直在用一个面向Firefox的插件，可以对页面性能评分。我们可以用类似这样的工具来检查。

Jim：听起来不错啊。

Frank：可以在移动浏览器上安装插件吗？

Joe：噢，你说到点子上了。我的手机上根本没办法安装插件。我喜欢的一些开发工具都是浏览器插件。没有这些工具，我们怎么能知道到底发生了什么呢？

Jim：前几天Kim向我展示了如何通过网络路由器的日志页面来监视对WiFi网络所做的每一个请求。我们能不能利用类似这样的方法，来监视手机做了些什么？

Frank：这个想法太妙了。不过，我们可以使用一个代理服务器。这与Kim向你展示的很类似，而且代理服务器正是为这种用途而设计。如果把手机关联到一个代理服务器，就能看到手机的所有Web请求。



YSlow插件可以根据Web页面如何构造给出一个性能评分。可以从这个网址得到YSlow (yhoo.it/yslow)。性能是每一个开发人员（尤其是移动Web开发人员）要关心的问题。

既然提到这一点，我们来花点时间讨论性能。不用担心，很快就会谈到移动优先的响应式Web设计，相信我，现在说的所有这些关于性能的内容并不是无关紧要的。

服务员，可以点菜吗？

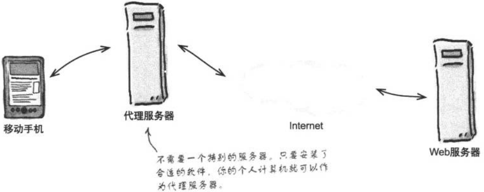


一个著名运动员在与对手的一场重要比赛前一晚因食物中毒住院了。警方怀疑是故意伤害，请一位侦探查清此事。侦探迅速行动，开始询问最佳目击者：服务员。

服务员拿着点菜单，写下顾客点的菜，把它交给大厨。菜做好后，服务员把菜端出来。一切她都看在眼里。

大多数情况下，在Web上你都像是在与大厨直接交谈。你与要访问的页面所在Web服务器之间什么联系也没有。

不过，如果有一个类似服务员的代理服务器，它会记录你要了什么，浏览器提供了些什么。现在我们可以做自己的侦探，找出事情真相。



我能找到一个代理来建立我的代理吗？

如果你做过很多移动方面的工作，就会发现学习如何建立一个代理服务器很有用。这是了解手机和服务器间发生了什么的最佳途径。

遗憾的是，建立代理服务器可能相当困难。好在，Blaze（一家移动性能公司）的一些好心人已经建立了一个免费的服务，除了安装你自己的工具外，这可能是最好的办法了。

速度不快时怎么办

Blaze提供了Mobitest，这是一个使用真正的iPhone和Android手机提供的免费移动性能测试服务。可以从www.blaze.io/mobile找到Mobitest。



Blaze的Mobitest就像是一个代理服务器。你告诉它你想测试哪个Web页面URL，另外想用什么设备来测试。Mobitest就会把你的测试请求放在对应该设备的一个队列中。

你请求的那种手机可用时，Blaze会跟踪记录它的测试手机与Web服务器间的所有通信，使你能了解到底发生了什么。

甚至还有一个有意思的特性，它能录制页面加载的一个视频，从而能看到使用这个手机的人所看到的情景。



试一试

来做些侦探工作，准备好了吗？现在来查看为什么On Tap Now页面这么慢。

1. **在www.blaze.io/mobile上测试On Tap Now页面。**

   On Tap Now 页面的地址是http://hf-mw.com/ch2/chapter2/ontap.html。

2. **查看加载时间和页面大小。**

   加载时间可以告诉你测试期间在这个手机上加载这个页面花费的时间。页面大小是与这个页面关联的所有资源的总大小，包括HTML、CSS、JavaScript、图像、字体等。

3. **尝试两个不同的手机，并比较速度。**

   不仅网络速度会有变化，手机本身处理和显示页面的速度也会不同。

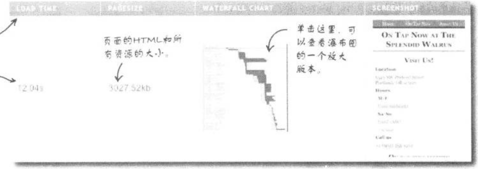
不要被外表迷惑，这实际上是一个很大的页面

这就对了！从测试可以看到，页面的大小大约有3MB。甚至对于一个桌面浏览器来说这都要算是一个相当大的页面了。在移动手机上难怪慢得像头大象。

实际上，对于桌面浏览器这也太大了。不能因为桌面屏幕大就以为连接速度快。任何地方都存在性能问题。

也难怪Mike的顾客对这个页面颇有怨言。在一个测试iPhone手机上加载这个页面居然要花10秒的时间。

使用iPhone得到的Blaze Mobitest结果页面。



取决于网络条件，加载时间可能有很大变化。Blaze是在WiFi上做的测试。

页面的HTML和所有资源的大小。

单击这里，可以查看瀑布图的一个放大版本。



什么是瀑布图？我看过了，不过并没有看出来是什么原因导致页面这么大。这个图到底有没有用啊？

瀑布图是一种常见的Web性能报告。

瀑布图会显示浏览器从服务器请求来构建Web页面的所有文件。直条表示下载一个资源花费的时间长度。这里按浏览器从服务器请求的顺序列出了相关的资源。

不过，不要"追逐"Blaze报告页面上的瀑布。这里并没有我们侦探工作需要的细节。接下来会告诉你如何找到一个更有用的瀑布。

HAR山里有黄金

Blaze Mobitest结果页面里藏着黄金呢，就在那个小小的“View HAR file”（查看HAR文件）链接下面。

单击这个链接，你会进入一个新网站，名叫HTTP Archive Viewer（HTTP归档阅读器），在这里可以看到一个比测试结果页面更详细的瀑布图。

这个瀑布图会为我们显示浏览器下载的每一个资源，这里会比Blaze报告页面上的瀑布图详细得多。

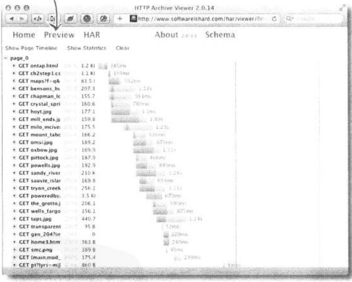
Blaze Mobitest结果页面上半部分。



在Twitter和Facebook链接旁边可以找到“View HAR file”（查看HAR文件）链接。

预览标签页中包含我们测试的这个页面的瀑布图。它由一个HAR文件生成。

HTTP Archive(HAR)阅读器。



HAR文件可以用来生成瀑布图。关于HAR请访问httparchive.org，在那里可以读到更多相关资料。

HAR表示HTTP Archive（HTTP归档）。这是一个文件规范，提供了一种标准方法来记录浏览器从服务器请求Web页面时发生了什么。

BRAIN POWER

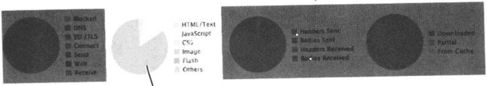
你能从这个瀑布图看出下载的文件中哪些可能有问题吗？

1000英尺的高空俯视：显示统计信息

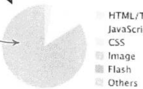
HAR瀑布图会显示你下载了哪些文件，服务器如何响应，以及下载花费的时间。不过在深入研究这个瀑布图之前，下面先从高层了解页面的统计情况。

单击HAR阅读器页面上的Show Statistics（显示统计信息）链接，可以看到一组共4个饼图。对我们来说最重要的是第2个饼图，它按文件类型对页面进行分解。把鼠标停在各个文件类型上，看看总的文件大小分别是多少。

单击HAR阅读器页面上的Show Statistics（显示统计信息）链接查看饼图。



哇！共有35个图像，单单图像就有2.4MB。

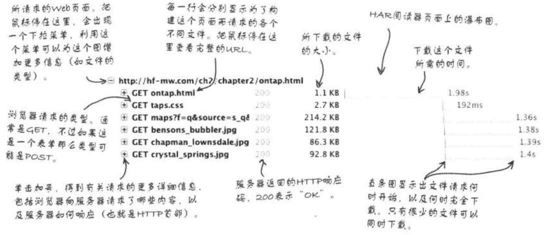


对应这个页面的8个JavaScript文件。所有JS文件的总大小是351 KB。



找出影响页面速度的累赘

现在来深入分析这个瀑布图，明确这些大图像和JavaScript究竟来自哪里。下面给出读这个图的要点。



浏览器和服务器之间的通信量可能极大，不过不要担心。你只需要注意两个问题：哪些资源最大，另外JavaScript来自哪里？

Exercise

再来看On Tap Now页面的瀑布图。找出5个最大的文件并分析。对于每个文件，请回答下面的问题：

1. **文件是什么类型？**
2. **文件来自哪个域？**
3. **如果文件是图像，则高度和宽度分别是多少？**

   提示：可能需要复制这个图像URL，在一个新的标签页中打开，或者直接下载这个图像来查看大小。

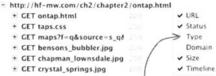
这些信息能告诉你什么呢？这对于你确定怎样做才能让页面更快会有什么帮助？

Exercise Solution

你发现页面的问题了吗？下面来分析。

1 文件是什么类型？

把鼠标停在文件名上，查看完整的URL和扩展名，这就可以确定文件的类型，还可以增加一个文件类型列，这样能快速扫描列表。

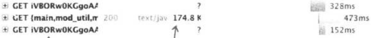


把鼠标停在页面URL上，单击URL旁边的下箭头，增加文件类型列。

2 文件来自哪个域？

现在到重点了。查看这个庞大的174.8 KB的JavaScript文件究竟来自哪里。

这个大文件并不是这里唯一可疑的文件。这些文件都有奇怪的名字。



把鼠标停在文件名上，查看完整的URL。

174.8KB，这是这个页面上最大的JavaScript文件。



Maps.gstatic.com是Google Maps的域。浏览器在下载一个地图的JavaScript，这个地图并未在移动屏幕上显示。很多可疑的文件都与Google Maps有关。

3 如果文件是图像，则它的高度和宽度分别是多少？

找出文件大小最大的图像。复制URL，并在一个新窗口打开。甚至不用看高度和宽度，也可以发现这些图像比在小屏幕上显示时大多了。



750像素高

682像素宽



168像素高

153像素宽

在iPhone上使用缩小的图像。

桌面和移动浏览器都用这同一个文件(121KB)，尽管图像显示大小不同。

*图像没有缩放。

Google Maps JavaScript从哪里来？

在移动手机上查看On Tap Now页面时，页面上并没有包含地图。为什么会下载这个JavaScript呢？下面在桌面浏览器中打开这个页面研究研究。

嘿，地图在这里呢。Mike肯定是做了特殊设置，使它只在较宽的屏幕上才会显示。

在移动手机上隐藏地图确实有些道理。这对于小屏幕来说太大了。较早的手机可能根本无法处理地图复杂的JavaScript，而且我们已经看到地图存在大量开销。

那么，Mike是怎样隐藏地图的呢？



嘿，神出鬼没的地图！

一行代码就会全部下载

地图是通过一个iframe包含在页面中的。这个iframe会加载建立地图所需的所有组件。

在ontap.html里找到这行代码。

```
<iframe id="map" width="300" height="300" frameborder="0" scrolling="no"
marginheight="0" marginwidth="0" src="http://maps.google.com..."></iframe>
```

这个iframe会导致下载47个文件！

这个URL实在太长了，这里做了省略。

Mike用CSS隐藏地图

Mike知道我们如何使用媒体查询针对移动设备修改布局。他在我们的媒体查询中加入了他自己的CSS规则。Mike增加的规则将iframe的display属性设置为none。

遗憾的是，将display设置为none后，尽管地图不再显示，但这并不能阻止它的下载。

taps.css

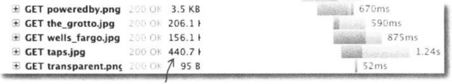
```
@media screen and (max-width:480px) {
  .
  .
  .
  #map {display:none;}
}
```

CSS文件中还有更多规则。

iframe的id为map。这个规则将display设置为none，这会隐藏Google Maps iframe。

这些大图片怎么处理？

这个页面上的图像需要减减肥了。下面来看瀑布图，找出最大的图像，看看它们为什么这么大。



taps.jpg文件有440.7KB，这是页面上最大的文件。



不过，这个超大文件是页眉图像，它根本不在移动设备上显示。

没错，但这并不意味着不下载这个文件。

类似于Google Maps的处理（CSS中display属性设置为none），页面中也隐藏了taps.jpg图像。不过，就像地图一样，设置display:none并不会阻止内容的下载。



```
@media screen and (max-width:480px) {
  [Other CSS rules are here]
  .header {display:none;}
}
```

这就是taps.jpg。在一个桌面浏览器中查看On Tap Now页面，可以看到哪里用到了这个图像。

流式图像都是超大图像

从瀑布图还可以注意到一点，啤酒标签都是大文件，大小在93到132 KB之间。它们都是桌面图像，只是使用我们在第1章学到的流式图像技术缩小了尺寸，以便放在屏幕上。

16个啤酒标签的总大小大约是2 MB。想办法优化这些图像是让页面速度快起来的关键。

所以这不是一个新问题，只不过页面上只有一两个图像时，这一点可能注意不到。但是一旦一个页面上放置16个标签，突然之间，这些流式图像就会成为让页面速度减慢的罪魁祸首。

看起来是移动友好的，但实际上不是

Jim：嗯，太郁闷了。我觉得外表真能糊弄人，是不是？

Frank：至少可以确认Mike还没有破坏这个页面。

Joe：是啊，我们任何人都可能犯同样的错误。这个页面上的问题相对严重一些，不过我想网站中每一个页面都可能存在同样的问题。

Frank：那我们该怎么办？为移动设备建立一个完全不同的网站版本？把响应式Web设计抛至脑后吗？

Joe：我们先不要太冲动。总有办法的。成功已经近在咫尺了。图像和JavaScript问题是不是什么共同点？

Jim：看起来所有问题都归结于一点，我们都是从面向桌面的内容开始（不论是图像还是地图），然后再隐藏这个内容。

Frank：完全正确。默认地会在浏览器中显示大文件，然后使用CSS将它们掩盖起来。不过，如果不仔细，那么看起来移动设备还是会下载这些大文件。如果某个方面出了问题，那么这可不是理想的选择。

Joe：如果反过来，那么默认发送最小的文件呢？

Jim：呵，有意思。这也是可行的。先从移动模板开始，然后为桌面浏览器增加内容。

Frank：你说的听上去很像那种渐进增强。

Joe：你说得对。这些年我们一直在使用渐进增强。现在唯一的差别在于我们是从移动版本入手，渐进地增强文档来满足桌面浏览器。

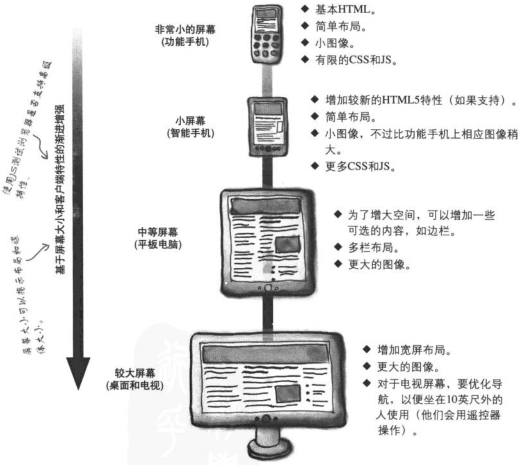
Jim：听上去应该可以。我们试试吧。



渐进增强可以促进分层Web页面的构建。至少，每个人都能看到和使用内容。如果有更强大的浏览器，那么可以得到更多样式层和交互性来改善用户的体验。

移动优先响应式Web设计

移动优先响应式Web设计（Responsive Web Design，RWD）的含义不言自明：这是一种从移动模板入手的RWD技术。尽管很简单，但这种方法确实能力非凡。



*这些只是一些增强的例子。你能做什么取决于具体项目。

什么是渐进增强？

渐进增强把Web设计看作是一系列分层。第一层是内容。结合内容和语义标记来创建结构化内容。如果在这里止步，那么你会得到一个几乎世界上所有浏览器都能读的文档。

有了这个基础之后，可以使用CSS增加一个表示层，再使用JavaScript增加一个行为层。绝对不要假设浏览器支持这些特性，不过倘若浏览器确实支持，访问者可以得到更好的体验。

多年来，Web开发人员通常都是先建立只在最高级浏览器上才能工作的应用，然后尝试确保这些Web页面能够在较早的浏览器上妥善降级（也称为优雅降级）。渐进增强则正好反过来。



渐进增强就像是一个分层蛋糕。蛋糕，真馋人。

移动优先设计的好处

移动优先RWD与渐进增强并没有不同。认识到这一点，所以很多人都把它称作是内容优先设计，因为内容正是渐进增强的第一层。

不论把它叫作什么，关键都是从最基本的文档开始，这一点不仅能保证让更多的人看到，还有一些很好的副作用。

移动优先有点像小碟节食法。用更小的碟子吃东西，这样往往会吃得更少。

桌面主页是一桌丰盛的大餐。各种各样的东西都在上面。

移动版本则是一个小碟子。你必须精心挑选，优先选择你的内容。

一旦有了一个涵盖重点的移动网站，接下来最好问一些基本问题做好准备，比如，如果有些内容在移动版本中没有出现，那么这些内容是不是确实重要，有没有必要再增加到桌面版本中。

语义标记特写

语义标记是指表达内容含义的HTML标记和属性。

例如，页面上<h1>标记包围的内容会比由<h2>或<p>标记包围的内容更重要。

如果class和id属性有类似calendar的值，而不是像left或top这样的表示值，这些属性也可以为文档增加语义含义。很多Web开发人员都采用一种标准方法使用类来提供更多语义含义，这称为微格式（microformat）。有关内容可以查看www.microformats.org。

语义标记并不表示完全不使用类似<div>和<span>这种没有增加含义的标记。事实上，要尽可能为页面的内容选择正确的语义标记和属性。

下面把这个Web页面反过来

因为我们已经在使用RWD技术，所以调整页面做到移动优先不用花费太长时间。下面给出一个简短的列表，列出了我们要做的几处修改。

- [ ] 让HTML尽可能简单，调整CSS的顺序，使移动版本最优先。 ← 稍后会解释为什么这样做。
- [ ] 调整CSS背景图像，对于每个图像只下载一个文件。确保适当地使用display:none。
- [ ] 对于不同屏幕分辨率，为<img>标记提供不同的源文件。确保下载大小适当的图像。
- [ ] 如果浏览器支持Google Maps，而且文档足够宽能够放下地图时，则用JavaScript为页面增加Google Maps。

On Tap Now页面的当前结构

打开chapter2目录中非凡海象网站的ontap.html文件。这个文件看上去与我们在第1章中构建的文档非常相似：

```
<div class="navigation">...</div>
<div class="header">...</div>
<h1>...</h1>
<div id="visit" class="column">...</div>
<div id="ontap" class="column">...</div>
<div class="footer">...</div>
```

两栏而不是三栏。

不再是名为"main"的<div>。Mike创建了一个新栏，id为"ontap"。这个<div>包含当前酒水列表。

因为我们已经用语义标记很好地创建了一个模板，这个文档已经很整洁也很简单。看起来要让内容做到移动优先，我们的主要任务就是去除Google Maps。

由于以后还需要引用这个代码，下面用HTML注释使这个iframe不再包含在页面中。

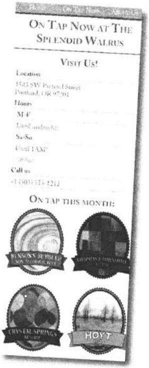
用<!--和-->包围Google Maps iframe，把它注释掉。

找到#visit <div>中的iframe。

```
<!--
 <iframe id="map" width="300" height="300" frameborder="0" scrolling="no"
marginheight="0" marginwidth="0" src="http://maps.google.com..."></iframe>
 -->
```

我看到的真是一个新页面吗？





问得好。内容的顺序现在还有问题。

在主页上，如果移动窗口显示的第一个内容是Visit Us（请访问我们）信息，那倒没什么。不过如果每个页面上Visit Us内容都重复出现，除非访问者向下滚动，否则肯定无法区别页面是不是有变化。

我们需要重排内容的顺序，让On Tap Now信息在Visit Us内容之前出现。

动手做！

复制id为visit的<div>中的所有内容，把它们粘贴到ontap <div>下面。

```
<div class="navigation">...</div>
<div class="header">...</div>
<h1>...</h1>
<div id="ontap" class="column">...</div>
<div id="visit" class="column">...</div>
<div class="footer">...</div>
```

BRAIN POWER

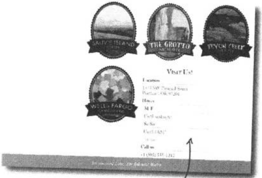
Visit Us内容对这个页面重要吗？更好的做法是不是该把它移到另外一个页面上，再链接到那个页面呢？或者干脆从移动页面中将它去除，如果是在一个更大的屏幕上显示页面，那么可以再用JavaScript增加这个内容，这样是不是更好一些？

修正内容浮动

我们对内容顺序的修改会破坏桌面浏览器中的布局。Visit Us部分现在位于页面最下面。

第1章中，我们曾经提到把右栏放在主栏前面是一种常用的技巧，这样可以更容易地处理布局中的浮动。如果你希望一个内容块浮动到某个内容旁边，就需要在源代码中把它放在前面。

不过不用担心。这里可以做一个简单的修正。之前我们是把Visit Us内容浮动到On Tap Now内容的左边。实际上，完全可以把On Tap Now内容浮动到Visit Us内容的右边。



Visit Us浮动到啤酒标签下面。

打开taps.css，完成以下两处修改。

之前

```
@media screen and (min-width:481px) {
  .column {
    margin: 10px 1.04166667% 0 0;
  }
  #visit {
    width: 31.25%;
    float: left;
  }
  #points {
    width: 25%;
    float: right;
  }
  #main {
    margin: 10px 27.0833333% 0
26.0416667%;
  }
  #ontap {
    margin: 10px 0 0 32%;
  }
}
```

把这些……（#visit 中的 width: 31.25%; float: left;）……改为这样。

把这些……（#ontap 中的 margin: 10px 0 0 32%;）……改为这样。

之后

```
@media screen and (min-width:481px) {
  .column {
    margin: 10px 1.04166667% 0 0;
  }
  #visit {
    margin: 0 68.75% 0 0;
  }
  #points {
    width: 25%;
    float: right;
  }
  #main {
    margin: 10px 27.0833333% 0
26.0416667%;
  }
  #ontap {
    width: 67%;
    float: right;
    margin: 10px 0 0 0;
  }
}
```

用67%而不是68.75%，这样可以为各栏提供一点点活动空间。

移动优先媒体查询

现在来做一些常规的工作。第1章中，我们先从一个桌面网站开始，然后把它改为适用于移动设备。现在我们把这个过程反过来，先从最简单的内容开始，逐步建成为桌面应用（而且会更棒）。



CSS会沿着小屏幕到大屏幕的路径层叠。

不过，首先需要承认，在CSS领域，移动优先的说法有些用词不当。对小屏幕应用媒体查询之前，我们要设置所有基本样式（包括颜色、字体等），然后再逐步增强。

这样做有一个很好的理由。很多移动浏览器根本不理解媒体查询。所以我们要确保它们至少能得到基本样式规则。

合理组织你的CSS

CSS文件通常就像是厨房的杂物箱。开始时可能很有条理，很整洁，不过过一段时间就会变得乱七八糟。要加入移动优先媒体查询，可能需要先从布局规则中挑出基本样式规则。

幸运的是，第1章建立的CSS已经很不错。大多数基本样式规则已经放在文件最前面，增加布局和格式的媒体查询位于文档靠后的位置。我们要做的就是把移动媒体查询放在桌面查询前面。

动手做！

```
/* Wider viewports/higher resolutions (e.g. desktop)  */
@media screen and (min-width:481px) {
  [Desktop layout rules here]
}
/* Mobile/lower-resolution devices */
@media screen and (max-width:480px) {
  [Mobile layout rules here]
}
```

把移动媒体查询块放在桌面媒体查询上面。这样一来，可以确保CSS的层叠效果与移动优先渐进增强方法的效果一致。

试一试

我们已经对页面做了一些修改，包括：

- 删除Google Maps
- 修正浮动
- 重排标记
- 重排媒体查询

最好做个检查，确保页面能正常工作。在一些桌面和移动浏览器中加载这个页面，看看外观怎么样。务必要在Internet Explorer中检查。

意外！Internet Explorer中页面出问题了



之前提示过你可能想在Internet Explorer (IE)中测试这个页面，不要告诉我那时你没有想到可能会出现这种结果。与IE斗争简直是Web开发人员的必经之路。你可能已经被先前的斗争吓住了，很清楚肯定会有一个IE怪兽在前面等着我们。

那么这到底是怎么回事？原因就在于IE不支持媒体查询。

先别把书丢到一边，责怪我们不该教一些在全世界最流行的浏览器中无法派上用场的东西，深呼吸，放松一点。有很多方法可以绕过IE的（很多）缺陷。

我们有点太武断了。IE9和更高版本确实支持媒体查询，所以情况会有好转。

由于IE 8及之前的版本不支持媒体查询，IE无法得到创建各栏的CSS规则。

Internet Explorer的救生艇：条件注释

Microsoft提供了一个很好的工具，可以帮助Web开发人员通过条件注释编写专门针对Internet Explorer的代码。

这里查看浏览器是否小于(lt)IE 9而且不是IE Mobile (!IEMobile)。这里排除了IE Mobile，因为这种浏览器能得到移动布局。IE9及以上版本理解媒体查询，所以它们不需要额外的帮助。

可以在http://bit.ly/ie-comments查看条件注释的完整语法。

```
<!--[if (lt IE 9)&(!IEMobile)]>
<link rel="stylesheet" type="text/css" href="layout.css" media="all" />
<![endif]-->
```

仔细观察。HTML注释在第一行，不过直到最后一行包含了"-->"这个注释才算结束。其他浏览器会把它看作是一个注释，并忽略其内容。

如果满足条件，IE会完成开始[if]语句和结束[endif]语句之间的所有工作。这个例子显示了一个CSS文件的链接，不过这可以是任何适合放在HTML文档中的内容。

对媒体查询使用条件注释

你可能注意到了，条件注释指向layout.css。现在就来创建这个文件。

我们要从当前样式表中取出一些规则。之所以把这个新文件命名为layout.css，这是因为它只用于有足够屏幕空间可以支持多栏布局的浏览器。

1 创建一个空的文本文件，命名为layout.css，把桌面规则复制到这个文件中。

一定要复制开始和结束媒体查询之间的所有内容，不过不要复制@media规则本身。

复制之后，从taps.css中删除这些规则和包围这些规则的媒体查询，以后还会对HTML文档应用这些规则。

taps.css

```
/* Wider viewports/higher resolutions
(e.g. desktop)   */
@media screen and (min-width:481px) {
  .column {
    margin: 10px 1.04166667% 0 0;
  }
  #visit {
    margin: 0 68.75% 0 0;
  }
  #points {
    width: 25%;
    float: right;
  }
  #main {
    margin: 10px 27.0833333% 0
26.0416667%;
  }
  #ontap {
    width: 67%;
    float: right;
    margin: 10px 0 0 0;
  }
}
```

把这些规则复制到你的新文件中。

layout.css

```
  .column {
    margin: 10px 1.04166667% 0 0;
  }
  #visit {
    margin: 0 68.75% 0 0;
  }
  #points {
    width: 25%;
    float: right;
  }
  #main {
    margin: 10px 27.0833333% 0
26.0416667%;
  }
  #ontap {
    width: 67%;
    float: right;
    margin: 10px 0 0 0;
  }
```

2 **增加一个指向这个新样式表的链接。**

对于支持媒体查询的浏览器，如果屏幕足够大，则我们会增加一个指向这个新layout.css文件的链接。

```
<link rel="stylesheet" type="text/css" href="taps.css" />
<link rel="stylesheet" type="text/css" href="layout.css" media="all and
(min-width: 481px)" />
```

把这个link标记增加到ontap.html。

min-width的值是481px，这是由之前从taps.css删除的媒体查询复制得来的。

这是第1章学过的link标记的媒体查询语法。

3 **增加IE条件注释。**

我们已经让它在大多数桌面浏览器上都能正常运行。现在只需要增加之前创建的条件注释就大功告成了。

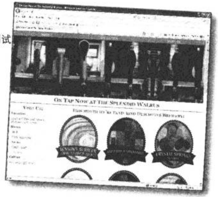
```
<link rel="stylesheet" type="text/css" href="taps.css" />
<link rel="stylesheet" type="text/css" href="layout.css" media="all and
(min-width: 481px)">
<!--[if (lt IE 9)&(!IEMobile)]>
<link rel="stylesheet" type="text/css" href="layout.css" media="all" />
<![endif]-->
```

条件注释重复上面这一行，确保桌面IE浏览器可以看到我们的layout.css文件。

4 **又该测试了。**

在支持媒体查询的浏览器以及不同版本的Internet Explorer中测试这个页面。看起来还不错，是不是？

即使是我们挑剔的老朋友IE现在也能正确地显示布局了。



there are no Dumb Questions

问：如果用业界最高性能的4G手机，性能还算是问题吗？

答：当然了。即使是4G手机，有时也可能会在EDGE网络中使用（EDGE是一个比较老、比较慢的网络）。研究表明，速度慢的网站会减少使用量，直接影响最终结果。

问：为什么我会从Blaze Mobitest得到不同的结果？

答：出现这种情况有很多原因。每一次测试时，取决于网络流量，页面下载时间会改变。对应各个操作系统的Google Maps代码都有所不同，而且可能随时间改变。随着操作系统新版本的发布，手机的行为也可能改变。在这本书中，我们使用了Blaze的iOS 4.3、Android 2.2和Android 2.3测试设备来完成测试。

对于测试结果的差别不用太过担心。重点是那些本来没有必要下载的代码和图像。

问：把样式表分解到两个文件中，这样不是使网站加载更慢了吗？

答：HTTP请求数会使下载速度有很大差别（请求数越多，下载越慢），这一点确实不假。所以我们不能莽撞地增加请求。在这里，我们认为分为两个文件更合适，这样IE也能使用同样的文件。

问：你提到过建立一个代理服务器可能很有意义。你有什么建议吗？

答：有很多代理服务器，包括一些非常棒的开源产品。很凑巧，我们是一个名为Charles Proxy的商业产品的粉丝。

问：缺少插件好像是个大问题。如果没有Firebug和Web Inspector，那么该怎么办呢？

答：这可不容易。首先，很多调试工作可以在桌面浏览器中完成，只要注意这个过程中有些时候要在实际设备上测试。

还有很多新工具在努力绕开这些插件限制。Mobile Perf Bookmarklet (http://bit.ly/mw-perf)就包括很多性能工具。weinre (http://bit.ly/mweinre)和Opera Dragonfly (http://opera.com/dragonfly)允许在桌面浏览器中运行Web Inspector来检查手机浏览器中发生的情况。

问：切换到移动优先媒体查询后看起来没有多大改变啊。那又何必这么麻烦呢？

答：对于这个页面，桌面优化CSS文件和移动优先CSS文件之间确实没有太大差别。不过，从我们的经验看，这只是个例外。如果有更复杂的样式，那么通常你会希望适用较宽屏幕的规则覆盖一些（但不是全部）为较小屏幕设置的样式。重排媒体查询顺序可以确保CSS层叠行为与随着屏幕变宽渐进增强页面的目标是一致的。

问：看起来桌面和移动浏览器中内容的顺序通常会不同。更复杂的页面中如何处理这个问题呢？

答：哈，你抓住问题的关键了，是不是？没错，这正是响应式Web设计中常见的难题之一。CSS3中的Flexible Box Module (Flexbox)承诺提供一种简便方法，可以在样式表中重排内容的顺序。通过结合Flexbox和媒体查询，你完全可以根据需要重排页面中的顺序。遗憾的是，Flexbox还只是初出茅庐，没有得到充分支持。所以开发人员会求助于JavaScript重排内容顺序，或者结合使用RWD和设备检测（见第5章）。坦率地讲，内容排序和图像处理仍然是RWD中的两大难题。

问：不支持媒体查询的IE版本能看到响应式设计吗？媒体查询是不是没有必要呢？

答：Internet Explorer会显示桌面版本。它还会显示流式内容和灵活的图像，不过不会基于媒体查询指令改变。如果媒体查询支持很重要，则可以使用一个名为Respond.js的开源库，它能为较早的IE版本提供媒体查询支持。这是一个相当全面的脚本，所以如果决定使用则一定要全面地测试。

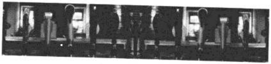
情况怎么样了？

我们已经建立了基本CSS，而且适当地调整了CSS的顺序。接下来要做什么？

- [x] 让HTML尽可能简单，调整CSS的顺序，使移动版本最优先。
- [ ] 调整CSS背景图像，对于每个图像只下载一个文件。确保适当地使用display:none。
- [ ] 对于不同屏幕分辨率，为<img>标记提供不同的源文件。确保下载大小适当的图像。
- [ ] 如果浏览器支持Google Maps，而且文档足够宽能够放下地图时，则用JavaScript为页面增加Google Maps。

调整页眉图像taps

从我们的瀑布图可以看出，有一个很大的CSS背景图像，不过利用display:none把它隐藏起来了。尽管它从来不在页面上显示，但浏览器还是会下载这个图像。



还记得我们的老朋友taps.jpg吧，移动设备会下载这个图像，但是从来不会在页面上显示。

因此要确保只在需要时才下载这个图像。怎么做到呢？可以把它放在一个媒体查询中，这样只有当屏幕宽度大于480像素时才会下载。

不过，并不是创建一个全新的媒体查询，完全可以把这些CSS规则放在layout.css中，这个文件已经通过<link>标记中的一个媒体查询包含在页面中。

把这些代码行从taps.css复制到layout.css的末尾。

```
.header {
   background:URL('images/taps.jpg') repeat-x;
   height: 300px;
}
```

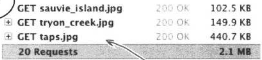
增加到layout.css之后，从taps.css删除这几行。



试一试

使用Blaze Mobitest检查On Tap Now页面，确保不再下载taps.jpg。要在iPhone和Android设备上都试一试。如果你的页面不在一个公共服务器上，则可以使用http://hf-mw.com/ch2/ex/3/ontap.html来完成测试。





Android看上去还是在下载taps.jpg。

Blaze的Mobitest指出，Android仍在下载这个图像，不过这只是一个误报。

Blaze必须修改它的手机来完成远程测试，这会导致一些偶然的奇怪行为。

使用一个普通的Android手机时，并不会下载taps.jpg图像。

图像优化卷土重来

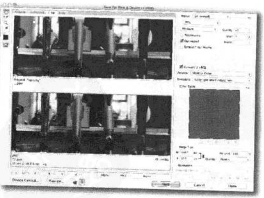
回想Web发展初期，Web开发人员很多时间都在考虑图像优化的问题。随着带宽的增加，Web开发人员不再像从前那样“斤斤计较”，竭力提高图像的每一分性能。不过，移动设备使图像优化再次变得重要。

看起来我们可以利用一些基本的Web图像优化技术让taps.jpg图像更小一些。可以把JPEG质量从80改为45，这样能让这个图像变成78KB而不是原来的440KB，而且必须仔细查看才能看出二者图像质量的差别。

实际上应该说图像优化一直以来都极其重要。更快的连接从来都是无法保证的，即使是桌面计算机也是如此。

要更多地了解Web图像优化，可以参见《Head First HTML with CSS & XHTML》的第5章。

图像还不够小？可以在www.smushit.com上进一步缩小。



将taps.jpg的优化版本从extras文件夹复制到images文件夹，用一个较小的版本替换原来的大文件。

我们使用Photoshop优化这个照片，不过完全可以使用你喜欢的其他Web图像编辑器。

一个src完全控制

CSS图像简直带来了一场图像之灾。每个响应式设计中<img>标记都可能带来问题，因为src属性只能有一个值，而不论屏幕大小如何。那么如何提供大小适当的图像呢？

On Tap Now页面上有16个啤酒标签，都使用<img>标记，比如这个Benson's Bubbler标签。

我们需要根据屏幕大小提供这个图像的多个版本，但HTML只允许src有一个值。

```
<img src="brews_images/bensons_bubbler.jpg" alt="Benson's Bubbler">
```

人们可能希望使用JavaScript替换src属性的值。遗憾的是，大多数浏览器会提前查看HTML文档，JavaScript可能还没有完全计算就已经完成了图像的预加载。通常这意味着在JavaScript改变src之前已经下载了大小相同的文件，这会导致重复下载，而且还会让浏览器重建页面布局。

也不能用CSS覆盖src属性的值。

响应式图像服务器可以帮忙

如果浏览器无法向服务器请求合适的图像，服务器就必须自己来确定了。这正是Sencha.io Src要做的事情。

这个调整图像大小的服务原来名为TinySRC。

我们可以使用Sencha.io Src为每个设备提供最佳大小的图像。

将src的第一部分设置为http://src.sencha.io/。

替换为你的图像域和路径。

然后增加你想要调整的图像的完整URL。

```
<img src="http://src.sencha.io/http://[DOMAIN]/[PATH]/brews_images/bensons_
bubbler.jpg" alt="Benson's Bubbler">
```



Sencha.io Src只会收缩图像。

它不会放大图像。放大图像会导致质量很差。最好是找到一个更高质量的源图像。

Sencha.io Src会调整图像大小以适合设备屏幕的大小。例如，如果用iPhone访问这个网站，图像就会限制在其屏幕大小（320 480像素）范围内。

动手做！

更新ontap.html文档中的所有啤酒标签图像，改为使用Sencha.io Src。

Sencha.io Src如何工作

Sencha.io Src的工作就像有魔法一样。你从iPhone请求一个图像，它会给你一个适合iPhone大小的图像。使用功能手机呢？没问题，Sencha.io Src会给你一个很小的图像。如果从桌面浏览器请求会怎么样呢？则它会提供原样大小的图像。

Sencha.io Src怎么知道要提交多大的图像呢？它会使用浏览器的user-agent串（这是每个浏览器都会提供的一个标识符），在一个大数据库中查找这个设备。这个数据库包含了成千上万设备的有关信息。这个设备数据库记录的信息之一就是屏幕的大小。

> user-agent串是什么？下一章会更详细地讨论。

一旦Sencha.io Src知道屏幕大小，则它开始将图像缩小到这个设备的最大宽度。然后把它创建的图像在一个缓存中存储30分钟，这样以后请求同样大小的图像时速度会更快一些。

> 如果需要Sencha.io Src提供一个特定的图像大小，则它也能做到。具体怎样做请查看http://bit.ly/senchasrc。

<img>标记没有最佳解决方案

使用Sencha.io Src也有缺点。它依赖于设备检测，有时这会出问题（第5章将更详细地讨论）。另外还需要通过一个第三方传送你的所有图像。

> 为什么<img>标记这么难处理：可以参考有关响应式图像的系列文章了解更多信息：http://bit.ly/rwdimgs1。

事实上，对于如何处理不同屏幕大小的不同图像源问题，目前并没有最佳的解决方案。不过，请紧密关注这个领域，因为很多人都在尝试寻找一个更好的解决方案。

最后的修改：优化的啤酒标签图像

与taps.jpg图像类似，通过保存JPEG质量稍有降低的啤酒标签图像，可以减小这些图像的文件。不用担心，我们会为你完成这些优化。可以在extras/labels-optimized目录中找到优化后的图像。

把优化后的图像复制到brew_images文件夹，替换所有现有的图像文件。

试一试

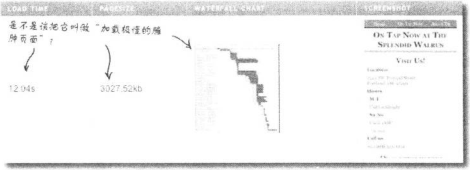
现在我们应该有了一个高效、快速、移动优化的Web页面。使用www.blaze.io/mobile测试这个页面，看看我们的工作成效如何。适当选择，让Blaze在手机上运行3次测试，得到平均值。将新页面的总文件大小和下载时间与原页面做个比较。

如果你的页面不能公开访问，则可以使用http://hf-mw.com/ch2/ex/4/ontap.html来测试。

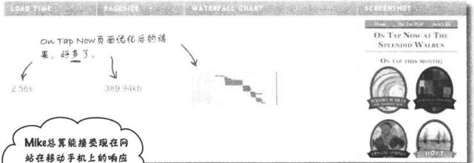
真是超快地移动Web页面

我们的减肥计划起作用了！现在On Tap Now页面比从前瘦了87%。iPhone下载时间从大约12秒减为3秒。

之前



之后



Mike总算能接受现在网站在移动手机上的响应速度了。



BRAIN POWER

Web性能优化是一个不断发展的领域，还有很多其他方法可以让页面速度更快。我们还可以采取哪些性能改善措施来提高这个页面的速度？

there are no Dumb Questions

问：为什么浏览器会下载根本不会用到CSS图像？

答：浏览器通常无法确切地知道一个图像并不会用到。这可能是一个只在某个JavaScript或CSS活动触发时才显示的图像。浏览器会提前下载这些图像，这样如果一个活动突然触发要显示的图像，访问者就不用等待了。

问：知道了，那么为什么不会下载媒体查询中的图像呢？

答：原先浏览器也会下载媒体查询中的图像。浏览器制造商了解了开发人员如何使用媒体查询，所以相应地调整了浏览器行为。这些都还是相当新的内容，正因如此，有些浏览器仍然会下载媒体查询中的资源（尽管并没有用到）。

问：通过Sencha.io Src传送图像安全吗？这让我很紧张。

答：小心一些当然很有必要。如果你的网站中某个重要部分集成了一个第三方服务，倘若这个服务出了问题，你就会受到牵连。

Sencha宣称专门致力于提供这个服务，而且会一直免费。不过可以想见，如果某个大型网站开始使用这个服务，则Sencha肯定需要得到相应报酬，否则将不能运行这个服务。

如果你不喜欢Sencha.io Src，则也可以使用我们在第5章介绍的设备检测工具构建一个类似的服务。

问：除了Sencha.io Src外，还有没有其他选择？对于<img>标记问题，有没有仅在客户端的解决方案？

答：在响应式设计中还有很多不同的方法来处理<img>标记。很多工作正在进行当中，力求找到一种不需要设备检测的解决方案。每种方案都有一些弱点，包括我们在这个项目中使用的方案。可以在 http://bit.ly/rwdimgs2了解这些技术的一个全面介绍。

问：其他媒体呢？视频和音频也存在同样的问题吗？

答：简单地讲，确实如此。HTML5视频和音频格式要稍好一些，因为它们允许你采用不同的文件格式定义媒体的“默认”版本作为退路。如果浏览器不支持提供的第一种选择，就会查看下一个选择。

尽管要好一些，不过这种方法对于解决网络速度或分辨率没有任何作用。在无线网站上使用移动手机的人可能不需要一个高清质量的视频。实际上，Apple公司的QuickTime视频提供了一个视频参考格式，可以根据互联网连接速度提供视频。

问：是不是只有这些？或者响应式Web设计是不是还有很多相关的未知情况和问题？

答：确实有很多难题。与所有新技术一样，人们仍在探索哪些可行，而哪些不可行。RWD很前沿。正因如此，我们在这本书中会谈到很多技术。通常你需要结合这些技术来提供最佳体验。

尽管存在这些问题，RWD的承诺还是激发着很多人在努力构建更完备的解决方案。RWD领域可谓日新月异。

放大,放大,砰……



实在抱歉!我真不愿意在你工作到一半时又提出新需求,不过我最好的顾客之一看那些小文本很困难,抱怨她没办法放大页面。你能修正这个问题吗?

还记得第1章中的viewport <meta>标记吗?现在来更详细地讨论这个标记。

viewport <meta>标记告诉浏览器期望的页面大小和级别(即放大级别)。其中还包含一些控制,可以避免用户改变页面的大小。

在viewport <meta>标记上控制放大

在ontap.html文档的<head>中可以找到viewport <meta>标记。语法相当简单。

这是什么类型的<meta>标记?

content属性包含一个逗号分隔的浏览器指令列表。可以在http://bit.ly/metaviewport查看所有选择。

视窗的宽度。可以按像素设置,也可以设置为"device-width",这会告诉浏览器按设备分辨率调整视窗。

```
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1" />
```

设置页面的初始级别(或放大级别)。把它设置为1,这表示文档应当按其正常大小显示。

对于如何缩放页面声明一个限制。还有一个类似的minimum-scale设置。

可以利用maximum-scale避免用户放大页面。

放大的权利？



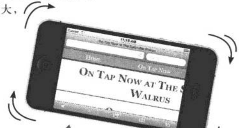
看起来能够放大页面对于页面的可访问性很重要。为什么有人会关闭这个特性呢？

这确实会影响可访问性。一些Web开发人员甚至声称移动设备上允许放大页面是一项基本权利。

我们没有必要那么夸张，不过放大确实很重要，在禁用之前一定要仔细考虑。

至于设计者为什么会禁用缩放，这有很多原因。如果页面在使用复杂的触摸手势，禁用放大后，人们能够更容易地在屏幕上成功划扫。

iOS中还有一个bug，设备旋转到水平模式时会改变放大等级。这个bug会让页面放大，导致页面的右边被切掉。



旋转一个iOS设备时，有一个bug会导致页面不再适应视窗大小，会把内容的右边部分切除。

再打开放大特性

要想再打开放大特性，需要从viewport <meta> 标记删除maximum-scale设置。

动手做！

编辑viewport <meta> 标记，删除"maximum-scale=1"。

```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

打开放大特性后，旋转iPhone或iPod Touch，查看iOS的放大bug究竟会导致怎样的结果。

一定要删除"initial-scale=1"后面多余的逗号。

重回到我们的正常日程

解决了突发的视窗问题之后，下面来看我们的进展情况。现在这个移动页面速度很快，已经很接近移动优先RWD。剩下要做的就是如果屏幕足够大再把地图增加到页面中。

- [x] 让HTML尽可能简单，调整CSS的顺序，使移动版本最优先。
- [x] 调整CSS背景图像，对于每个图像只下载一个文件。确保适当地使用display:none。
- [x] 对于不同屏幕分辨率，为<img>标记提供不同的源文件。确保下载大小适当的图像。
- [ ] 如果浏览器支持Google Maps，而且文档足够宽能够放下地图时，则用JavaScript为页面增加Google Maps。

使用JavaScript增加地图

现在只剩下一项任务，就是在浏览器窗口足够宽时再把地图增加到页面上。我们已经看到，如果使用CSS隐藏和显示地图，则地图的资源还是会下载。

所以需要使用JavaScript在适当的时候增加地图。可以把这看作是我们已经了解和喜欢的媒体查询的一个JavaScript版本。

得到之前预留的Google Maps iframe代码。现在需要把它放回到页面中，以便显示地图。下面将更仔细地分析这个iframe代码。

还记得之前注释掉的这个iframe代码片段吗？现在要使用JavaScript把它插入到页面中。

```
<!--
 <iframe id="map" width="300" height="300" frameborder="0" scrolling="no"
marginheight="0" marginwidth="0" src="http://maps.google.com..."></iframe>
 -->
```

考虑再三，地图还是很有用的

位置，位置，位置！

谈到移动手机，即使是句老话也会有新的含义。由于很多手机可以通过GPS或者其他三角定位方式确定你所在的位置，使用位置提供更有关联性的内容是很常见的。

Mike为移动设备隐藏了地图，因为地图实在太大了。我们已经看到为显示地图需要下载多少个文件，所以隐藏地图是有道理的。

不过这并不是说地图不好。所以在比较窄的屏幕上可以不嵌入地图，而是链接到地图。

增加一个指向地图的链接

要链接地图，需要一个<div>，我们的JavaScript可以引用这个<div>。它包含一个<p>标记，其中包括一个地图链接。你能想出我们为什么要这样安排吗？

动手做！

需要一个让JavaScript引用的容器，所以在这里增加一个<div>。

利用<p>标记的id，对于较窄的屏幕我们可以在链接上面插入iframe。稍后还会介绍更多有关内容。

把这个新<div>增加到注释掉的iframe代码上面。

```
<div id="mapcontainer">
  <p id="maplink">
    <a href="http://g.co/maps/jm2pw">View Google Maps</a>
  </p>
</div>
<!--
  <iframe id="map" width="300" height="300" frameborder="0"
scrolling="no" marginheight="0" marginwidth="0"
src="http://maps.google.com..."></iframe>
  -->
```

在JavaScript中建立一个伪媒体查询

下面来看用来在页面中插入iframe的JavaScript代码。这个代码就相当于一个非常简单的媒体查询。

这个变量对应容器元素的id，我们希望把地图增加到这个容器中。这里使用了一个变量，这样将来可以更容易地修改<div>。

将breakpoint变量设置为481像素。breakpoint（分界点）表示一个临界宽度，屏幕达到这个宽度时就会向页面增加地图。

查看视窗宽度是否大于breakpoint。

增加一个新的iframe元素，把它赋至mapElement变量。

这些代码行为这个新iframe元素增加所有属性。属性及属性值都是从Google Maps iframe代码片段复制得来的。

Google Maps提供的URL很难看。你可能不愿意自己重新键入。可以在extras/map.js中找到这个代码。

最后这一步把iframe (mapElement)增加到mapcontainer <div> (id)中，放在包含链接(maplink)的段落前面。

```
<script type="text/javascript">
var breakpoint = 481,
  id = 'mapcontainer',
  viewportWidth = window.innerWidth;
if (viewportWidth > breakpoint) {
  var mapElement = document.createElement('iframe');
  mapElement.id = 'map';
  mapElement.width = '300';
  mapElement.height = '300';
  mapElement.frameborder = '0';
  mapElement.scrolling = 'no';
  mapElement.marginheight = '0';
  mapElement.marginwidth = '0';
  mapElement.src = 'http://maps.google.com/maps?f=q&so
urce=s_q&hl=en&geocode=&q=334+NW+11th+Ave,+Portland,+O
R+97209&aq=&sll=37.0625,-95.677068&sspn=58.164117,80.3
32031&vpsrc=0&ie=UTF8&hq=&hnear=334+NW+11th+Ave,+Portl
and,+Oregon+97209&t=m&ll=45.525472,-122.68218&spn=0.01
804,0.025749&z=14&output=embed';
  document.getElementById(id).insertBefore(mapElement,
maplink);
}
</script>
```

去除注释掉的iframe代码

我们不再需要原来的iframe代码，所以将它从HTML文档中删除。

删除这些代码行！

```
<!--
<iframe id="map" width="300" height="300" frameborder="0" scrolling="no"
marginheight="0" marginwidth="0" src="http://maps.google.com..."></iframe>
-->
```

将JavaScript增加到On Tap Now页面

现在我们要把这个JavaScript增加到页面。由于地图是一个锦上添花的特性，而不是基本要求，所以我们打算把它作为最后才增加的一个内容，到最后才让浏览器把它增加到页面上。

打开ontap.html，找到HTML文档的末尾处。

把JavaScript作为最后一个内容增加到页面上，放在结束</body>标记前面。

```
<div class="footer">
  <p>See you soon! Love, The Splendid Walrus</p>
</div>
[INSERT SCRIPT HERE]
</body>
</html>
```

把不太重要的JavaScript放在页面最下面，这是一种很好的做法，可以让页面加载得更快。浏览器会解析所有HTML和CSS，然后才会到达JavaScript。访问者能更快地得到一个可用的页面，而不用呆呆地等待地图代码加载。



该测试一下我们的工作了。对于这个ontap.html，回答以下问题：

1. **会在移动手机上下载这个JavaScript吗？**

   在iPhone和Android上使用www.blaze.io/mobile/加载这个页面。检查瀑布图，查看是否在下载Google Maps代码。如果你的web页面不在一个公共网络上，则可以使用http://hf-mw.com/ch2/ex/5/ontap.html来测试。

2. **在较大的屏幕上地图会出现吗？**

   在你喜欢的桌面浏览器中打开On Tap Now页面。窗口宽度大于480像素时地图会出现吗？

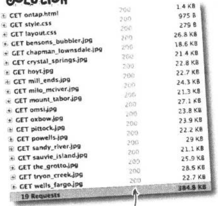
3. **地图适应响应式设计吗？**

   试着调整浏览器窗口的大小。地图会像设计的其他部分一样相应缩放吗？地图显示有没有什么问题？

Exercise Solution

看起来怎么样？有问题吗？

1 会在移动手机上下载这个JavaScript吗？



看吧，没有Google Maps下载了。

使用iPhone利用Blaze Mobitest服务检查这个页面。单击结果页面上的HAR文件链接得到详细的瀑布图，查看下载的所有文件。

地图文件没有下载。太棒了！

那么Android情况怎么样呢？Blaze指出JavaScript还是下载了，不过这也是一个误报。

我们提到过，Blaze必须修改它的手机以便完成远程测试。这个修改有一个奇怪的副作用，就是在测试手机上运行的JavaScript报告的屏幕宽度会远远宽于(800像素)未修改的手机报告的结果。

相信我们，这个JavaScript能正常工作，另外Android上并没有下载地图代码。

2 在较大的屏幕上地图会出现吗？

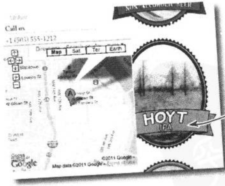
是的。在我们的桌面浏览器上地图看起来很不错。

地图出现在Chrome中，这说明我们的JavaScript起作用了。



3 地图适应响应式设计吗？

哦，不妙，这里有问题了。地图没能像响应式设计的其余部分那样相应缩放。不仅如此，屏幕上有些地方地图还覆盖了啤酒标签。



哎呀。浏览器窗口变窄时，地图会覆盖标签。

为什么地图不能像页面上的图像那样缩放呢？

这些部件不是响应式的



响应式Web设计实在是个新兴产物，所以像Google Maps之类的部件默认情况下不太可能是流式的。

公司提供部件嵌入到其他Web页面中时，他们会尽其所能确保这个部件能正常工作，而不论页面有怎样的布局。这通常意味着HTML本身会对类似高度和宽度等属性提供硬编码设置。

我们需要处理不完善的第三方部件，这几乎是每一个移动网站都要面对的问题。响应式设计还有一个额外的需求，就是部件也应当采用流式设计。

宽度和高度是固定的，这就阻止了地图的缩放。

```
<iframe id="map" width="300" height="300" frameborder="0" scrolling="no"
marginheight="0" marginwidth="0" src="http://maps.google.com..."></iframe>
```

这个iframe上的很多属性都可以移至CSS。

理想情况下，HTML只包含内容和标记。它不应包含任何表示信息。

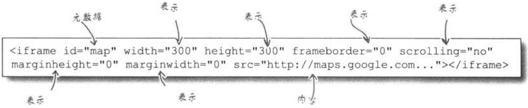
BRAIN POWER

哪些CSS属性与iframe中使用的属性对应？

将iframe属性移至CSS

下面尽可能将iframe的属性移至CSS，并使它们采用流式设计。

首先，需要创建想要移动到CSS的属性列表，找出哪些属性对应表示，而哪些属性对应内容或元数据。



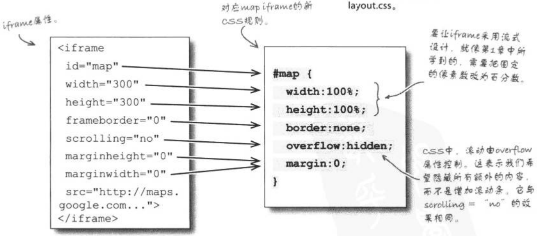
所有这些表示属性都应该放在CSS中。

样式/属性配对

有些属性与相应的CSS属性同名。不用太费劲就能找到CSS中的宽度和高度属性（width和height）。另外一些属性，如frameborder，就不那么明显了。幸运的是，总的来讲CSS属性还是相当简单的。

动手做！

把这些规则增加到layout.css。



从JavaScript删除属性

既然已经由CSS来提供表示，下面修改我们的JavaScript，不再设置那些表示属性。删除之前找到的那些增加表示属性的代码行。

找到ontap.html最下面的JavaScript。

```
<script type="text/JavaScript">
var breakpoint = 481,
  id = 'mapcontainer',
  viewportWidth = window.innerWidth;
if (viewportWidth > breakpoint) {
  var mapElement = document.createElement('iframe');
  mapElement.id = 'map';
  mapElement.width = '300';
  mapElement.height = '300';
  mapElement.frameborder = '0';
  mapElement.scrolling = 'no';
  mapElement.marginheight = '0';
  mapElement.marginwidth = '0';
  mapElement.src = 'http://maps.google.com/maps?f=q&so
urce=s_q&hl=en&geocode=&q=334+NW+11th+Ave,+Portland,+O
R+97209&aq=&sll=37.0625,-95.677068&sspn=58.164117,80.3
32031&vpsrc=0&ie=UTF8&hq=&hnear=334+NW+11th+Ave,+Portl
and,+Oregon+97209&t=m&ll=45.525472,-122.68218&spn=0.01
804,0.025749&z=14&output=embed';
  document.getElementById(id).insertBefore(mapElement,
maplink);
}
</script>
```

从JavaScript删除这几行代码。（指向 mapElement.width 至 mapElement.marginwidth 六行，已划掉）

试一试

保存layout.css和ontap.html。在Safari中加载On Tap Now页面。
如果想更方便，则可以使用http://hf-mw.com/ch2/ex/6/ontap.html。
地图看上去怎么样？

现在找到酒吧应该毫无问题了

地图有点过于跋扈了，是不是？它几乎占据了整个左栏。它就像在高调地唱着“我是地图，我是地图”，以便引起你的注意。除非我们来驯服它，让它低调一点。

如果把窗口变窄，就会了解为什么地图希望得到我们的关注。地图会挤压成一个细长条，直到最后完全不可用。

我们还是希望地图能缩放，不过需要对各个方向的缩放程度设置一个界限。

这些截屏图都取自Safari。其他浏览器的表现会有所不同。不过无论哪个浏览器，地图都不像我们期望的那样。



1 地图的height属性值太大了。

将height设置为100%，这会使地图比页面上最高的图像还要长。下面把height设置为400px，对它多加一些控制。

在#map CSS规则中增加这行代码。

```
height: 400px;
```

2 嘿，看呐，地图就像穿了紧身牛仔裤。

窗口变窄时，地图会变得太细，大多数信息都将无法看到。我们需要设置一个最小宽度，使地图不会变成一个细竹竿。

增加这两行新代码之后，layout.css中的#map规则会是这样。

```
#map {
  width:100%;
  height:400px;
  border:none;
  overflow:hidden;
  margin:0;
  min-width: 200px;
}
```

又出现地图重叠了

我以为之所以让iframe采用流式设计，根本原因就是要消除重叠。现在我们已经把相应的表示属性放在CSS中了，不过浏览器窗口变窄时地图还是会覆盖啤酒标签。



将iframe表示属性移到CSS中只是第一步。现在需要用全新的眼光考察我们的媒体查询。

第1章中，我们使用媒体查询在宽度为480像素时切换布局。这个宽度是根据流行智能手机的宽度确定的。

对于地图，有一点可以确定：要决定在哪里应用媒体查询，首先需要查看页面的内容。

窗口变窄时，地图再次覆盖了啤酒标签。



让内容成为你的向导

使用480像素作为媒体查询的分界点存在一个问题。并不是所有手机都有相同的宽度。即使如今大多数手机确实宽度相同，谁能保证将来其他宽度（如540像素）不会成为最常见的宽度呢？

一种更好的方法是，对布局做修改时，让你的内容做向导。

我们并不是要求你与内容联系，真正让它与你交谈。不过，如果调整了浏览器窗口的大小，有些地方看起来不太对劲时，则从内容可以了解到很多情况。

可能图像变得太小， 也可能分栏变得太窄。

发生这些情况时，你就需要一个分界点了。然后可以调整媒体查询，修改这个分界点对应的表示。



该拉伸浏览器了

需要适当安排我们的内容，为此要尽可能放大和缩小页面，同时观察布局何时会被破坏。

不过，在此之前，我们需要有办法知道页面出问题时屏幕有多大。对此最容易的方法就是安装一个bookmarklet，它会为你显示窗口大小。

bookmarklet是一小段JavaScript，存储在浏览器书签中。

在浏览器中安装bookmarklet

前往http://bit.ly/window-resize，把标为Window Size（窗口大小）的链接拖到你的书签工具栏上，就能创建 bookmarklet。单击bookmarklet将它激活。调整浏览器大小，观察浏览器窗口左上角的数字变化。



单击

Safari书签工具栏中的Window Size书签。

单击Window Size bookmarklet时，会在窗口的左上角显示窗口的大小。



可选：安装扩展包

还有一个面向Google Chrome的扩展包，不仅会显示窗口大小，还会调整窗口的大小，使之与常用的屏幕分辨率一致。这个扩展包可以在http://bit.ly/chrome-resizer得到。

Web Developer Toolkit (http://bit.ly/webdevtoolkit) 会在标题栏中显示页面大小，同时提供另外一组有用的工具。这个工具包适用于Firefox和Chrome。

Sharpen your pencil

在安装了Window Size bookmarklet（或浏览器扩展包）的浏览器中加载On Tap Now页面。激活bookmarklet。调整浏览器大小。

写出布局被破坏或者内容看起来很奇怪时的浏览器宽度。

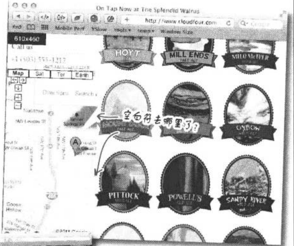
....

Sharpen your pencil Solution

下面来分析调整浏览器大小时出现的几个情况。

610像素：啤酒标签紧挨着地图。

大约610像素时，标签与地图挨上了。如果要创建一个新的分界点来解决这个问题，则需要在它们挨上之前完成。这说明，并不是使用610像素，而应该使用640像素作为分界点。





1200像素：庞大的标签。

浏览器越来越宽时，啤酒标签会变得异乎寻常的庞大。从多大开始算是过大呢？这是一个美学问题。对我们来说，浏览器宽度为1200像素时这些标签就显得过大了。

BRAIN POWER

调整浏览器大小时你还发现其他问题了吗？你认为问题要多严重才有必要另外增加一个媒体查询来解决？

分界点来帮忙

总的说来，还不算太坏。只需要对CSS做几个小小的改动就能解决问题。

收缩无比庞大的啤酒标签

目前每一行有3个啤酒标签。页面变得更大时，每行就会留出4个啤酒标签的空间。

为宽度超过1200像素的窗口创建一个媒体查询，将页面上一行的啤酒标签改为4个。

只有当窗口大于1200像素时才会做这个改变。

```
@media screen and (min-width:1201px) {
  .taplist li {
    width: 25%;
  }
}
```

将包含啤酒标签的列表项（li）的宽度设置为25%，这会在每一行上放置4个标签。

把这些规则增加到layout.css。

随着屏幕变宽，让图像成比例地缩小，这是很常见的做法。

很快变为单栏

即使啤酒标签没有与地图重叠，640像素时布局也会变得很拥挤。我们并不是增加一个新的分界点来解决这个重叠问题，可以把现在的媒体查询从480像素移至640像素。

我们的HTML、JavaScript和CSS都引用了480像素，所以这三者都需要更新。

做这个改变会把布局转换为一个单栏布局，并隐藏地图。这还有一个附加的好处，可以对大于480像素的手机应用单栏布局。

ontap.html

```
<link rel="stylesheet" type="text/css" href="layout.css"
media="all and min-width: 641px)">
```

将min-width设置为641px。

```
<script type="text/
javascript">
var breakpoint = 641,
  ...
</script>
```

将JavaScript中的breakpoint设置为641px。

taps.css

```
/* Mobile/lower-resolution devices
*/
@media screen and (max-width:640px)
{
```

将max-width设置为640px。



我们的移动优先响应式设计已经大功告成了！

- ☑ 让HTML尽可能简单，调整CSS的顺序，使移动版本最优先。
- ☑ 调整CSS背景图像，对于每个图像只下载一个文件。确保适当地使用display:none。
- ☑ 对于不同屏幕分辨率，为<img>标记提供不同的源文件。确保下载大小适当的图像。
- ☑ 如果浏览器支持Google Maps，而且文档足够宽能够放下地图时，则用JavaScript为页面增加Google Maps。

there are no Dumb Questions

问：到底什么是视窗？

答：假设拿一个纸板，在中间切出一个矩形。把这个矩形贴到你的监视器上，这样你只能看到Web页面上透过这个矩形的一部分。这就是Web页面的一个视窗。

问：那么viewport <meta>标记就是在告诉浏览器视窗的大小，是吗？

答：完全正确。默认地，iOS将视窗设置为980像素。如果你已经为较小的屏幕优化了页面，则设置<meta-viewport>标记会让浏览器知道要相应地设置视窗。

问：什么是分界点？

答：分界点只是描述分辨率的一种有趣的说法，达到这个分辨率时设计者会改变页面的布局。这通常通过媒体查询检查来完成，查看页面是否窄于或宽于某个像素数。

复杂的响应式设计可能有多个分界点，包括一些可能对布局全面修改的分界点，以及一些小的分界点（只做一些有针对性的调整来修正小的布局问题）。

问：我不希望阻止人们放大页面，不过这个iOS bug实在很讨厌。有没有办法允许放大而且不破坏页面呢？

答：可以在这里找到解决这个问题的一个JavaScript，https://gist.github.com/901295。

问：为什么刚开始会与地图重叠呢？

答：这是因为地图是一个不会随浏览器窗口缩放的元素。窗口很小时，浏览器不会把地图缩得更小，所以最后左栏会与右栏重叠。

问：向地图增加min-width难道不会破坏响应式设计，这样会创建一个不随浏览器窗口缩放的元素吗？

答：理论上是这样。不过这里看来是一个合适的解决方案，因为我们修改了媒体查询来解决内容重叠的问题。另一种选择是使用媒体查询调整地图的大小和分栏的比例。

BULLET POINTS

- 为一个已有的桌面网站增加媒体查询后，在移动设备上可能这个网站看起来还不错，但这并不意味着它是移动优化的。
- 因为大多数移动浏览器不支持插件，能为移动Web开发人员提供帮助的工具很少。
- 使用一个代理服务器或者类似Blaze Mobitest的测试方案，可以帮助你了解一个移动浏览器具体下载了什么。
- HTTP归档（HAR）文件和瀑布图是基本的性能工具。
- 移动优先响应式Web设计可以帮助优化Web页面，确保默认下载较小的资源。
- 移动优先RWD是渐进增强的另一种形式，使用屏幕大小来确定如何增强Web页面。
- 移动优先设计要求你关注真正重要的部分，相应地帮你从页面上去除不必要的东西。
- Internet Explorer 8及以下版本不支持媒体查询。条件注释可以作为一种解决办法。
- JavaScript能提升媒体查询，检查屏幕大小，并在适当的情况下增加内容。
- 不要根据屏幕分辨率设计分界点，应当由内容指示达到哪些分辨率时需要修改布局。

3 单独的移动网站

直面不太好的环境



响应式Web的远景固然美妙……

在这里，每个网站都采用一种移动优先的方法巧妙构建，只有一个布局，可以应对各种设备和环境。嗯……真不错。不过，如果现实一点，考虑一些实际因素会怎么样呢？比如说，遗留的系统、老式的设备，另外还有可能受到顾客预算的限制。有些时候，如果实在不能把桌面和移动支持完美地结合在一起，是不是需要单独来建设？这一章中，我们会介绍有关的细节问题，包括检测移动用户，支持那些老式手机，以及构建一个单独的移动网站。

Creature Comforts的现场代理处

Creature Comforts International（动物关怀国际组织）是一家全球性非盈利组织，着力帮助自然灾害灾区里那些生病或受伤的动物，为受到灾害影响的农民和牧民提供支持。直至目前，这家机构的代理处均依赖话音通信或临时的抗震笔记本电脑来协调人员和补给。



代理处如何得到和分享他们需要的信息?

Creature Comforts并不是一家新成立的组织机构，它的前身早在20年前就已经成立了。如今，这家机构已经拥有大量内部基础设施，包括一个重要的“传统”Web网站，这是基于一个财产内容管理系统（Content Management System，CMS）构建的。



对移动Web的更大需求

Creature Comforts的员工越来越发现，现场最可靠的（通常也是唯一可行的）连接渠道就是通过当地的蜂窝网络。地基互联网连接很难找到，不仅需要更多的地面设施，而且移动性也很有限。

Creature Comforts需要一个移动网站，一个能够支持多种设备、可以使用多种不同连接的网站。

听上去可是一个大工程啊。真是等不及了，现在就动手吧，让代理处的CMS发布更多移动友好的内容，另外还要使用一些响应式设计的东西。



Frank：我有一些坏消息。Creature Comforts没有太多预算，另外还涉及一些内部政策问题。而且，要想从原来那个财产CMS抽取出这个组织的管理和内容发布流程，可真要大费一番周折呢。我们还不能改动桌面网站，至少现在不能。

Jim：这么说，我们的任务岂不是无法完成了？

Frank：那倒不是，不过可能需要做一点让步。Creature Comforts的网站很庞大，也很复杂，不过在这个组织看来，要针对移动设备优化网站，最至关重要的部分就是他们所谓的"Comforts Logistics Portal"门户应用。这个Web应用允许代理处提供和接收更新信息，还可以协调日程和供给。桌面网站的这一部分内容相当全面，有一些API可供我们使用。

Jim：我不太明白。我们怎么能选择性地只改变网站的一部分呢？

Frank：在这里，我认为需要为移动用户开发一个单独的网站。

Jim：听起来真乱。

Frank：移动Web可能就是一个"脏活"。你知道的，我们要让派往世界各地的人使用这个移动网站，而且还要让网站可靠地运行。另外，这些人并不全都拥有最新的智能手机。很多员工的设备只是一些捐赠的老式手机，而且在他们服务的地区，移动连接的状况往往不太好，最好情况下也只能断断续续地连接。我们需要一个简洁但功能完备的移动网站，能帮助这些人完成他们的工作。不过不能简单地把现有桌面网站的全部内容与我们真正需要的东西混杂在一起。

有时为移动设备创建一个单独的独立网站更合适。

Relax

害怕编程？不要太担心。

这里提到了API和Web应用，不过不会让你做任何具体的编程。这些编程工作都留给我们，你的任务就是让网站看起来漂亮些，在移动Web上更好用。

there are no Dumb Questions

问：内容管理系统（Content Management System，CMS）用来做什么？

答：Web CMS是一个集编辑、发布和版权管理工具于一身的应用，用于创建和管理Web内容。有些CMS功能相当全面，为快速开发Web应用提供了一个环境［有时这也称为内容管理框架，或CMF（Content Management Framework）］。

管理员用户可能不熟悉HTML标记或Web设计，不过利用CMS，他们能轻松地创建和管理内容。

开源和商业领域都有一些CMS，可能用你能想到的任何一种语言编写。比如WordPress、Drupal、DotNetNuke、Joomla!和SharePoint。有时比较大或专业的组织会创建自己的CMS软件。

大多数CMS还会使用模板系统或其他机制处理内容的发布。由于这个原因，如果想通过转换来支持移动设备会很困难，因为在很多情况下，内容会与表示层夹杂在一起。Creature Comforts正是这种情况。它的CMS是很多年前设计的，只有一组模板。现在要从头开始重新建立系统对于这个机构来讲成本就太高了。

问：每一个Web CMS针对移动设备做适应性调整都很成问题吗？

答：现有的CMS系统实在太多了。其中一些针对移动设备调整会比另外一些容易。当然，很多流行的CMS都存在内容、逻辑和表示混杂在一起的问题。

很多CMS的开发群体和公司都在积极行动，开发优化的后续版本，可以将内容发布到不同类型的客户端。移动Web世界里一些高瞻远瞩的人正在重新考虑建立内容结构的方式，希望能更系统地处理内容（如应用数据），从而能在多个平台上更简单地实现其重用。

API特写

API（应用编程接口）是开发人员创建的有系统的、明确定义的接口，使不同的软件系统可以相互通信。Twitter API就是Web上一个很受欢迎的API例子。Twitter API定义了一组方法，Web程序员可以使用这些方法获取和修改Twitter系统中的数据。

Creature Comforts Web应用小组编写了网站的一部分，允许代理处和管理人员管理分布在各地的不同小组人员和供给。作为这个代码开发一部分，Web应用小组创建了一个API，可以用来获得和更新小组成员和物资的信息。

这个API会返回结构化数据，移动Web开发小组可以使用这些数据建立Creature Comforts网站的一个移动版本。

与CMS不同，Creature Comforts的API没有把逻辑和表示混在一起，这让我们的移动优化工作容易多了。

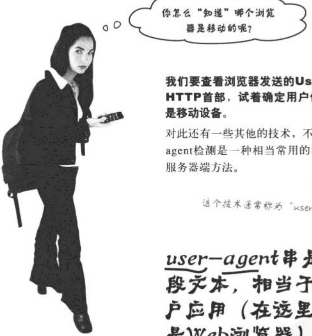
让移动用户前往移动优化网站

有时有必要为移动设备和桌面浏览器分别提供不同的网站。通常，你会希望用户只访问一个域（比如，www.example.com），然后根据他们的设备自动地路由到适当的网站。



寻找移动用户

要实现这种设置，也就是把移动设备重定向到移动优化网站（针对移动设备优化的网站），Web服务器需要知道进入的请求是否来自一个移动设备。



我们要查看浏览器发送的User-Agent HTTP首部，试着确定用户使用的是不是移动设备。

对此还有一些其他的技术，不过通过user-agent检测是一种相当常用的检测设备的服务器端方法。

这个技术通常称为"user-agent嗅探"。

user-agent串是一段文本，相当于客户应用（在这里就是Web浏览器）的身份证。每个不同的浏览器都是一个唯一的用户代理。

"用户代理"通常缩写为"UA"。

认识用户代理

Web浏览器（嗯，这也包括移动浏览器）发送一个Web页面或资源请求时，会发送一个User-Agent首部作为HTTP请求的一部分。

User-agent串很早就被网站管理员用来（当然也包括滥用）识别（以及错误地识别）浏览器。回想20世纪90年代那段一切都很混沌的日子，数百万用户都遭遇过臭名昭著的提示“该网站最好使用Internet Explorer浏览”（也可能是Mozilla、Netscape或者当时更流行的某种浏览器），其万恶之源正是user-agent嗅探。

```
GET /pretty.png HTTP/1.1
---------------------------------------------------------------
Host: www.example.com
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip,deflate,sdch
Referer: http://www.example.com/foo
User-Agent: Opera/9.64(Windows NT 5.1; U; en) Presto/2.1.1
---------------------------------------------------------------
```

请求行。

请求首部。

请求体（如果有）

HTTP请求的简化结构

user-agent历史分析

如今，user-agent串的结构很让人奇怪，有时甚至是惯例、混乱和欺骗拼凑在一起形成的乱七八糟的产物。关于如何写user-agent串从来没有人成功地给出过完备的说明。所以你可能会看到类似这样的东西：

```
Mozilla/5.0 (Windows NT 6.0) AppleWebKit/535.1 (KHTML, like
Gecko) Chrome/14.0.792.0 Safari/535.1
```

当然是……Windows Vista上运行的Chrome 14!

为什么Windows上的Chrome用户代理会提到Safari？什么是KHTML，另外它怎么“像”（like）Gecko？就像你的阑尾，或者是鲸残余的小粗腿，这算是一种进化的遗留产物。

“Mozilla兼容性标志”（上例中的Mozilla/5.0）并不完善，不过如今几乎无处不在，尽管它本身不再有什么实际意义。之所以提到Mozilla、KHTML、Gecko或WebKit，实际上是用户代理以这种方式声明其布局引擎相当于或者“优于”这些浏览器。写这本书时，移动设备上所有基于WebKit的浏览器（除了Android）都会在其user-agent串中提到“Safari”（Apple公司原来基于KHTML开发了WebKit）。

用户代理方式特写



下面来深入分析一些具体的user-agent串。

Windows Vista上运行的Chrome 14（第98页上已经见过）。

```
Mozilla/5.0 (Windows NT 6.0) AppleWebKit/535.1 (KHTML,
like Gecko) Chrome/14.0.792.0 Safari/535.1
```

```
Mozilla/4.0 (compatible; MSIE 7.0; Windows Phone OS 7.0;
Trident/3.1; IEMobile/7.0; SAMSUNG; SGH-i917)
```

运行Windows Phone 7的Samsung手机。

黑莓9700 (Bold)。

```
BlackBerry9700/5.0.0.442 Profile/MIDP-2.1 Configuration/CLDC-1.1
VendorID/612
```

分析组成

Windows NT 6.0
Windows Phone OS 7.0
BlackBerry9700/5.0.0.442

平台、操作系统和版本。

浏览器名和版本。

IE Mobile/7.0
Chrome/14.0.792.0

MSIE 7.0
Safari/535.1
KHTML, like Gecko
Mozilla/5.0
Mozilla/4.0

这个浏览器"兼容"或者历史上基于……

……其他元素

Profile/MIDP-2.1

面向移动设备的Java运行时库。

Configuration/CLDC-1.1

Java ME框架规范。

布局引擎（Trident是IE的布局引擎）。

AppleWebKit/535.1
Trident/3.1

VendorID/612

标识开发商/运营商。

SAMSUNG; SGH-i917

这个三星手机还发送了模型信息。尽管很常见，不过这并不必要。

WHAT'S MY PURPOSE?

1

```
Mozilla/5.0 (Linux; U; Android 2.1; en-us; ADR6200 Build/ERD79)
AppleWebKit/530.17 (KHTML, like Gecko) Version/4.0 Mobile Safari/530.17
```

2

```
Opera/9.64(Windows NT 5.1; U; en) Presto/2.1.1
```

3

```
Mozilla/5.0 (BlackBerry; U; BlackBerry 9800; en-US) AppleWebKit/534.1+
(KHTML, like Gecko) Version/6.0.0.246 Mobile Safari/534.1+
```

你已经见过一些分解的user-agent串。现在试试看，把从以上user-agent串中抽取的各个片段与相应的作用连线。

Presto/2.1.1	操作系统、平台和版本
Apple WebKit/534.1+	历史/兼容标志
Windows NT 5.1	浏览器名和版本
Mozilla/5.0	布局引擎
Version/6.0.0.246	历史/兼容标志
KHTML, like Gecko	布局引擎
Opera/9.64	浏览器版本

答案见第102页。

用户代理：魔鬼蛋

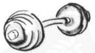


嗯，差不多吧。

如果在一个聚集着Web开发人员的房间里大声说“user-agent嗅探”，别人肯定会怪异地看着你，你会遭到奚落，甚至有些批评还会很尖锐。

user-agent嗅探把很多开发人员都引入歧途。我们已经看到，user-agent串会多么复杂。另外，user-agent欺骗也很常见，这是指用户（有意或无意地）把浏览器配置为发送一个不同的UA首部。而且如今有太多太多不同的UA，每一天还有更多新UA进入市场。

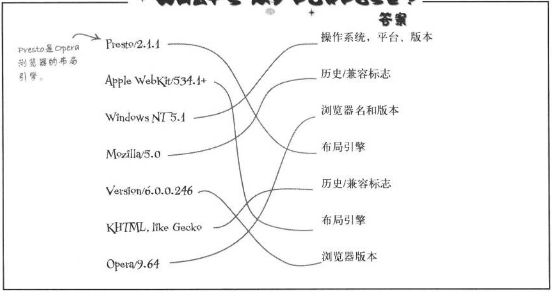
通过在服务器上嗅探用户代理来检测移动浏览器，这无疑是一种不高雅、不正确的行为。不过，有时这是完成这个工作的最佳（也是最原始的）工具。



BRAIN BARBELL

想想看尽管user-agent嗅探不太好，为什么有时却又是必要的呢？

WHAT'S MY PURPOSE? 答案



这是什么浏览器？

是不是对这个练习里用到的示例user-agent串（第100页）很好奇？下面向你完全解密，给出它们来自哪些客户应用：

1. Droid Eris（也叫作HTC Desire）Android 手机，运行Android 2.1浏览器。
2. Opera 9，运行在Windows XP上。
3. 黑莓9800 (Torch)，运行BlackBerry OS 6浏览器。

把第100页上的黑莓Torch (OS 6) UA与第99页上的黑莓Bold (OS 5) UA做个比较。是不是不一样？对于OS 6，黑莓浏览器使用WebKit布局引擎，它们的UA串看起来与其他WebKit浏览器更相似。



Watch it!

并不是所有user-agent串都遵循一致的模式。

有些user-agent串会更合理。用户代理并不都有同样的组成部分，有些完全是特立独行。大多数都会遵循一些基本模式，不过不要过于依赖这一点。别担心！我们有一些很好的工具来对付它们（起码是大多数）。

there are no Dumb Questions

问：第99页上user-agent串中你没有解释的那些部分有什么用呢？en-us是什么意思？U又是什么意思？

答："en-us"是语言信息，表示浏览器和界面元素是为美国英语本地化的。"U"指示浏览器有"强"安全性（与"I"不同，这表示"弱"安全性，"N"则表示根本不安全）。要分析user-agent串中类似这样的部分，可以访问一个很有用的网站：www.useragentstring.com。

问：是不是现有的每一个构建、发布、版本、平台、设备和开发商都有一个不同的用户代理？

答：确实如此。

问：除了浏览器还有其他应用发送User-Agent首部吗？

答：向Web服务器发出请求的很多其他类型的应用也会发送user-agent首部。这些应用包括（但不仅限于）搜索引擎、爬虫工具和各种Internet机器人（包括善意和恶意）。有时Web服务器日志会捕获到不正常地访问网站的有关user-agent信息，如构建到某个笔记桌面应用中的一个Web阅读器，或者是一个Web书签工具，也可能是一个监控服务ping连接到一个网站确保它可以操作。

问：为什么人们要"欺骗"浏览器的用户代理？

答：为什么有人会故意让他们的浏览器替他们"撒谎"，你是这个意思吗？这有很多原因。其中有两个常见的动因，一是因为隐私，二是对某个特定体验的需求。有些用户不希望别人了解他们使用哪个浏览器的有关信息。还有一些情况下，一个用户可能希望尝试以某种特定的方式查看Web内容。很多移动浏览器提供的"桌面模式"就是这方面一个流行的例子。如果用户打开这种模式，则通常（有时在不知道的情况下）会导致浏览器发送一个不同的user-agent首部，让它看起来更像是一个桌面浏览器。有时这会妥善地处理，不会有什么问题，Windows Phone 7桌面模式用户代理就很容易识别。不过并不总是如此。可以看看移动浏览器Skyfire的一个Android版本在桌面模式时发送的user-agent串：

```
Mozilla/5.0 (Macintosh; U;
Intel Mac OS X 10_5_7;
en-us) AppleWebKit/530.17
(KHTML, like Gecko)
Version/4.0 Safari/530.17
```

这看起来与桌面OS X Safari实在太相似了，这是因为这个用户代理确实完全等同于桌面Safari的一个用户代理。

问：不过如果有人在把他的移动浏览器伪装成桌面浏览器时遇到麻烦，我们不该尊重他的愿望，为他提供一种桌面体验吗？

答：这是一个棘手的哲学问题，我们很乐意回答"是"，填补与响应式设计之间的空白（也就是适应环境，而不论浏览器声称自己是谁，前提是浏览器足够现代，可以支持媒体查询之类的特性）。关于用户是否知道他们在做什么和没有做什么，还有些模糊的灰色地带，不过，嘿，就我们而言，你应该知道你要些什么。对于这一点，开发人员之间有很多分歧。这个话题有些沉重了，轻松点吧。

问：嘿，你完全没有提到UAProf。

答：很多移动设备发出请求时会发送UAProf（用户代理简档，User-Agent Profile）首部，通常（但不总是）作为x-wap-profile请求首部。UAProf规范为移动手持设备制造商提供了一种提供设备相关信息的方法。通常会在请求首部中提供指向UAProf位置的链接。

听上去很不错，不过遗憾的是，并不是所有设备和浏览器都发送UAProf首部。而且很大一部分指向UAProf的链接都是不合法的。另外，对于UAProf文件本身中数据的数量和质量也存在不一致。

问：OK，这么说来，有如此众多的用户代理，而且UAProf还不太可靠，这可真不妙。不过你说有一种明智、合理的方法可以完成基于用户代理的检测，是吗？

答：我们会在第5章了解一些组织和项目，它们的存在就是为了跟踪用户代理，建立关于用户代理的元数据以及类似UAProf等其他数据源的数据库。到那时，我们会力求简单，使用一个基本的服务器端脚本来查找具备移动浏览器用户代理主要特点的user-agent串。

实话实说：大多数主要网站都有一个单独的移动网站

世界上最大的网站大多都有一个单独的移动网站，而且其中大多数都使用某种形式的user-agent嗅探来完成路由。如果user-agent嗅探如此不堪，那么为什么还会出现这种情况？可见user-agent方法肯定有它的好处。

服务器端更自然……

把进入的信息流重定向到一个移动网站更适合在服务器端完成，这样更自然。如果把移动请求发送到一个单独的网站（对于Creature Comforts就需要这么做），则最好是在内容发送到客户端之前就做出决定。

……而且user-agent首部是最好的线索

user-agent首部尽管不算完美，但这是服务器关于请求浏览器所掌握的最直接也最可靠的线索。还有一些其他HTTP请求首部会提供客户“移动性”的提示，不过与这个让人喜忧参半的user-agent首部相比，那些HTTP请示首部都没有这么普遍。在一大堆不算完美的选择中，它算是最佳选择了。另外，尽管我们花了很多笔墨指出user-agent嗅探在哪些方面可能有问题（嘿，像是魔鬼代言人），但大多数情况下它还是能正常工作，特别是以可靠的方式使用时它会是不错的选择。



我们可以使用一个简单、免费的脚本来完成基本的移动设备检测。

如果你真正想要的是重定向

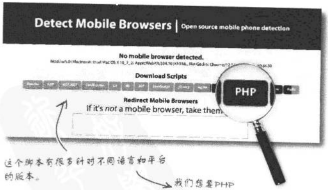


问得好。幸运的是，Frank和Jim能与Creature Comforts的IT部门合作。

我们能够在它现有的Web服务器上放置一个简单的重定向脚本。Creature Comforts的Web小组会把这个移动检测脚本放在每个页面最上面，这样移动信息流就能得到重定向。

查看脚本

现在来快速访问www.detectmobilebrowsers.com。在这里你会发现一个免费的、简单的移动检测脚本。我们希望你能找到最新版本，所以现在就去吧！一定要下载PHP版本。



看一眼脚本

```
<?php
$useragent=$_SERVER['HTTP_USER_AGENT'];
if(preg_match('/android|avantgo|blackberry|blazer|compal|elaine|fennec|hiptop|iem
obile|ip(hone|od)|iris|kindle|lge |maemo|midp|mmp|opera m(ob|in)i|palm( os)?|phon
e|p(ixi|re)\/|plucker|pocket|psp|symbian|treo|up\.(browser|link)|vodafone|wap|wi
ndows (ce|phone)|xda|xiino/i',$useragent)|| preg_match('/1207|6310|6590|3gso|4thp
|50[1-6]i|770s|802s|a wa|abac|ac(er|oo|s\-)|ai(ko|rn)|al(av|ca|co)|amoi|an(ex|ny
|yw)|aptu|ar(ch|go)|as(te|us)|attw|au(di|\-m|r |s )|avan|be(ck|ll|nq)|bi(lb|rd)|
bl(ac|az)|br(e|v)w|bumb|bw\-(n|u)|c55\/|capi|ccwa|cdm\-|cell|chtm|cldc|cmd\-|co(
mp|nd)|craw|da(it|ll|ng)|dbte|dc\-s|devi|dica|dmob|do(c|p)o|ds(12|\-d)|el(49|ai)
|em(l2|ul)|er(ic|k0)|esl8|ez([4-7]0|os|wa|ze)|fetc|fly(\-|_)|g1 u|g560|gene|gf\-
5|g\-mo|go(\.w|od)|gr(ad|un)|haie|hcit|hd\-(m|p|t)|hei\-|hi(pt|ta)|hp( i|ip)|hs\-
c|ht(c(\-| |_|a|g|p|s|t)|tp)|hu(aw|tc)|i\-(20|go|ma)|i230|iac( |\-|\/)|ibro|idea|
ig01|ikom|im1k|inno|ipaq|iris|ja(t|v)a|jbro|jemu|jigs|kddi|keji|kgt( |\/)|klon|kpt
|kwc\-|kyo(c|k)|le(no|xi)|lg( g|\/(k|l|u)|50|54|e\-|e\/|\-[a-w])|libw|lynx|m1\-
w|m3ga|m50\/|ma(te|ui|xo)|mc(01|21|ca)|m\-cr|me(di|rc|ri)|mi(o8|oa|ts)|mmef|mo(
01|02|bi|de|do|t(\-| |o|v)|zz)|mt(50|p1|v )|mwbp|mywa|n10[0-2]|n20[2-3]|n30(0|2
)|n50(0|2|5)|n7(0(0|1)|10)|ne((c|m)\-|on|tf|wf|wg|wt)|nok(6|i)|nzph|o2im|op(ti
|wv)|oran|owg1|p800|pan(a|d|t)|pdxg|pg(13|\-([1-8]|c))|phil|pire|pl(ay|uc)|pn\-
2|po(ck|rt|se)|prox|psio|pt\-g|qa\-a|qc(07|12|21|32|60|\-[2-7]|i\-)|qtek|r380|r60
0|raks|rim9|ro(ve|zo)|s55\/|sa(ge|ma|mm|ms|ny|va)|sc(01|h\-|oo|p\-)|sdk\/|se(c(\-
|0|1)|47|mc|nd|ri)|sgh\-|shar|sie(\-|m)|sk\-0|sl(45|id)|sm(al|ar|b3|it|t5)|so(f
t|ny)|sp(01|h\-|v\-|v )|sy(01|mb)|t2(18|50)|t6(00|10|18)|ta(gt|lk)|tcl\-|tdg\-
|tel(i|m)|tim\-|t\-mo|to(pl|sh)|ts(70|m\-|m3|m5)|tx\-9|up(\.b|g1|si)|utst|v40
0|v750|veri|vi(rg|te)|vk(40|5[0-3]|\-v)|vm40|voda|vulc|vx(52|53|60|61|70|80|8
1|83|85|98)|w3c(\-| )|webc|whit|wi(g |nc|nw)|wmlb|wonu|x700|xda(\-|2|g)|yas\-
|your|zeto|zte\-/i',substr($useragent,0,4)))
header('Location: http://detectmobilebrowser.com/mobile');
?>
```

detectmobilebrowser.php.txt

这就是脚本！你下载的脚本可能看上去与这里不完全一样（可能是一个更新的版本），不过应该是类似的。

Relax

目前这个脚本看起来可能像个野兽，不过不用花太多功夫就能把它驯服。

第一次在你喜欢的文本编辑器打开detectmobilebrowser.php时，你可能会发现它看上去有点……可怕。不过，不用害怕，我们会带你了解它是如何工作的，实际上，要让它起作用，你要做的只是几个小小的改动，我们会告诉你怎么做。

动手做！

将这个文件重命名为redirect.php，再把它保存到chapter3文件夹中。

这个脚本如何工作？

这个脚本会检查user-agent串，并确定它看上去是不是移动浏览器，如果是就进行重定向。共分3个步骤完成：

1. **获取请求浏览器的user-agent串。**

   在PHP中，可以从全局变量 `$_SERVER['HTTP_USER_AGENT']`访问得到。

2. **使用一些精心建立的正则表达式。**

   这里有两个正则表达式。它们都会分析user-agent串，并查看是否与已知的移动值匹配。你可能会发现一些明显的片段，如“hiptop” 或“symbian”。不过还有一些不那么清楚的部分，这些正则表达式是那些通晓移动世界的人开发的！

   正则表达式，也称作“regex”或“regexp”，是一种匹配串模式的形式方法。

3. **如果浏览器的user-agent串与任何一个正则表达式匹配，就完成重定向。**

   如果浏览器的user-agent串与任何一个正则表达式匹配，这个脚本会用指定的位置设置一个Location首部，重定向移动信息流。

   默认地，移动信息流会重定向到 www.detectmobilebrowser.com/mobile，我们要改变这个设置。



确保你的工作环境确实适合

继续之前先要做一些安全检查！下一节（以及本书很多其他部分）要求你有一个能正常工作的Web服务器，而且允许使用PHP。这可以是你自己的计算机，或者可以找一个托管提供商，我们可不挑剔。不过，如果你还没有一个适合的工作环境，现在就请翻到附录ii，找到建立工作环境的有关信息。

建立一个移动模拟网站



Jim：嘿，Frank，你怎么建立Creature Comforts的移动网站呢？

Frank：我想无非是编写代码使用他们的API，同时完成移动设计，所以我已经飞快地建立了一些简单的模拟网站，我们可以从它们起步。一次解决一个问题，好吗？

Jim：听上去不错。你使用响应式内容了吗？

Frank：嗯，差不多吧。我使用了比例宽度，不过我发现要处理一些年代比较早的手机，它们可能不能理解媒体查询和帮助实现响应式设计的其他特性。我得到了当前Agent Portal登录页的桌面HTML模板，进行简化，并重构CSS做到移动友好。我用到了一些示例数据，这样现在就不用担心功能、API等其他方面了。

Jim：移动设备检测和重定向脚本呢？怎么测试它们呢？

Frank：嗯，听说我们最后会把它放在Creature Comforts的服务器上，别人会从要提供的各个页面调用这个脚本。不过，对现在来说，我们希望在本地测试，检查重定向的表现如何。

对移动重定向脚本的几处修改

要对这个重定向脚本做几处定制修改，来了解它的实际使用。我们想增加一行新代码，定义如果脚本看到一个移动用户会重定向到哪个位置。PHP `$_SERVER['HTTP_HOST']`变量是运行这个脚本的Web主机的名字。

Web主机（例如，"www.example.com"，或者如果你在自己的计算机上，可以是"localhost"或"10.0.0.2"或者诸如此类的名字）。

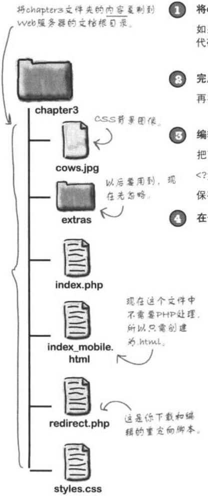
我们还希望修改发送Location响应首部的那行代码，想把移动信息流重定向到我们的移动模拟网站，而不是www.detectmobilebrowser.com网站。

就像请求首部一样（如User-Agent）HTTP响应（由服务器发回）也有首部。

```php
<?php
$useragent = $_SERVER['HTTP_USER_AGENT'];
$mobile_location = 'http://' . $_SERVER['HTTP_HOST'] . '/index_mobile.html';
if(preg_match(...))
  header('Location: ' . $mobile_location);
?>
```

redirect.php

试一试



1. **将chapter3文件夹的内容复制到Web服务器的根目录。**

   如果你不希望把这些文件放在文档根目录中，没问题。只是要确保更新代码，修改定义$mobile_location的那行代码中的路径。

2. **完成第108页上对重定向脚本的修改。**

   再在文本编辑器中打开这个重定向脚本（如果还未打开）。在脚本中完

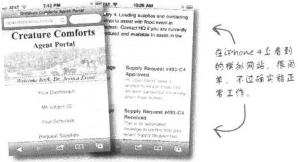
3. **编辑index.php。**

   把下面这行代码增加到index.php文件的最上面：

   ```
   <?php require _ once('redirect.php'); ?>
   ```

   保存这个文件。

4. **在一个移动浏览器（模拟器或真正的移动浏览器）中查看这个模拟网站。**



Relax

这些只是模拟的。

这一章要使用的移动模拟网站只是模拟的。它们使用示例数据来表现“插入”真实数据API时网站会是什么样子。这里表示的数据是虚拟的，很多链接并不起作用。不要担心！

复杂因素……的特别快递



嘿，Frank。我与为员工分发手机的小组谈过了，发现很多手机都在运行一个名叫Opera Mini的浏览器。我还给你快递了我们正在使用的一些常见手机，方便你做测试。

Opera Mini，是吗？

如今，黑莓WebKit浏览器和iOS Safari之类的顶尖浏览器笼罩在巨大的光环之下，不过不要低估Opera Mini！Opera Mini和Opera Mobile是姊妹浏览器，它们是全世界最流行的移动Web浏览器。

例如，2011年上半年，Opera就报告称有超过1.1亿用户。这相当于印度移动Web流量的一半以上，而且超过了尼日利亚移动Web流量的90%。

Opera Mini在连接速度很慢或者数据很昂贵或受限的地方尤其流行。

Opera Mini实际上是一个移动转码器。这表示，用户在Opera Mini中访问一个Web页面时，请求会通过Opera的服务器路由传送，在这里针对移动设备压缩并优化Web页面和资源。这个优化意味着最后向设备传送的字节会更少，从而得到一种更快的体验（通常也更便宜）。

Opera Mini设计为支持各种不同（通常是较低端）的手机，所以它与那些功能完备的智能手机浏览器表现有所不同。

Opera Mini不只是在发展中世界得到关注。它在美国和欧洲的功能手机中也相当流行。



动手做！

我们将在Opera Mini 4.2中查看这个移动模拟网站。幸运的是，Opera有一个很方便的基于Web的模拟器可供使用（这是一个Java Applet）。让你的Web浏览器指向：

http://www.opera.com/mobile/demo/?ver=4

Sharpen your pencil

下面来看这个移动模拟网站在Opera Mini 4.2模拟器中的表现。使用Web嵌入式Opera Mini模拟器查看http://hf-mw.com/ch3/ex/1/模拟网站。

看看哪里不同（或者，唉，看看哪里出问题），记录下来。在Opera Mini模拟器和iPhone或Android模拟器（第109页）中分别查看这个模拟网站，你能找到二者之间至少4处不同吗？

1 ....

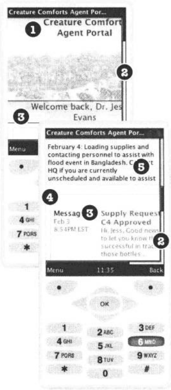
2 ....

3 ....

4 ....

Sharpen your pencil Solution

模拟网站在Opera Mini 4.2中看上去大不相同，是不是？下面来看看有所不同或者有些奇怪的几个地方。我们发现了5个有问题的地方。



1. <h1>和<h2>标记没有样式，而且非常小。由于这个原因，页眉中的背景图像看上去很傻。

   我们确实无法在Opera Mini中对任何标题元素加样式。

2. 整个页面太宽了，还有一部分从屏幕右边消失了。

   这个问题很突出。

3. 没有很好地应用文本格式。斜体文本没有真正斜体，粗体文本也不是粗体。

4. 没有显示圆角矩形，显示的是简单矩形。

5. 状态消息使用了overflow: scroll，这在所有移动设备上都是绝对应该避免的。内容被切掉，而且也不能正常滚动。

   如果你已经在iPhone或Android上注意到这一点，给你额外加分！

there are no Dumb Questions

问：为什么非得在http://hf-mw.com上查看这个例子呢？为什么不能用我自己的版本？

答：Opera Mini是一个代理浏览器。这说明，所有Web信息流都会通过一个Opera服务器路由传送。所以，这个示例模拟网站和其他代码必须托管到Opera服务器能到达的某个地方。

如果你在你自己的计算机上完成工作，则可能会有一个对整个互联网不可见的IP地址（所谓的内部IP地址，只对你连接的那个网络可见）。

如果你在一个托管服务器上工作，或者把Creature Comforts网站放在一个外部可以访问的Web服务器上，那么当然可以使用你自己的版本。

并不是所有手机都是智能手机……完全不是这样

Creature Comforts开展工作的很多国家都把智能手机看作是很稀缺的东西。在印度，2011年智能手机的销售量还不到所售手机总量的6%。你可能认为这没什么……是吗？

印度到底有多少部手机？

到2011年8月，印度有超过8.5亿移动手机客户。这使印度成为移动手机总数仅次于中国的第二大国。

尽管智能手机在印度还很少见，但是移动手机甚至比马桶还常见！

这绝不是开开玩笑。联合国发行了一个报告，比较了移动手机相对于廉价卫生基础设施易得到的程度。

如果我的用户不在印度呢？在美国，所有人都使用智能手机。我不用担心那些功能手机。



功能手机超越智能手机的情况并不仅仅出现在新兴市场。

实际上，直到最近美国的智能手机销售量才超过了功能手机。可能到2012年的某个时间，智能手机会最终代表美国的大多数手机。大多数欧洲国家也是这样。

看到智能手机爆炸式增长的报告时，一定要记住，即使有这样的增长，替换掉所有那些老手机仍然有待时日。

嘿，功能手机也没有那么糟

当然，它们的浏览器很老，很奇怪，而且通常连接状况不太好，这让我们很头疼，不过，功能手机确实让世界各国的很多人生活发生了巨大改变。

如果你想了解某个重大创新，则可以到有新兴市场的发展中国家旅游，观察他们如何使用移动手机来完成从农业到金融业的一切事务。

对于很多人来说，手机并不只是一个口袋大小、轻巧可爱的现代计算机。它是他们与世界联系的主要连接通道。

力求基本：认识XHTML-MP

再来分析index_mobile.html，你会看到这个模拟网站目前使用了这个DOCTYPE声明：

```
<!DOCTYPE html>
```

这是HTML5的DOCTYPE，在这个现代世界里，对于健壮的网站和Web应用来说，这是一个非常棒、很有吸引力的选择，不过如果我们想访问和支持Creature Comforts员工使用的那些移动设备，这个DOCTYPE就不一定正确了。现在来为你介绍我们的新朋友： XHTML移动简档（XHTML Mobile Profile，XHTML-MP）。

XHTML-MP是专门为支持中级和功能手机以及一些二流Web浏览器开发的一种XHTML。实际上，从它出现到现在已经好几年了，不过由于它那个更有魅力、更年轻的兄弟HTML5的飞速成长，XHTML-MP越来越被大家所忽视。但是对于像Creature Comforts移动网站这样的项目，它还是很有用的。

XHTML移动简档（XHTML Mobile Profile）是你了解并喜爱的桌面(X)HTML的一个子集。这是HTML的一个简化版本，设计为在各种不同的移动浏览器上都能很好地工作。很多移动浏览器都提供了XHTML-MP支持。



成品DOCTYPE

下面把模拟网页转换为一个XHTML-MP文档。编辑index_mobile.html文件，把HTML5 DOCTYPE （文件最上面一行）替换为下面这两行：

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.2//EN"
 "http://www.openmobilealliance.org/tech/DTD/xhtml-mobile12.dtd">
```



index_mobile.html

XHTML-MP是一种XML，所以我们还需要这个XML声明。

这是对应XHTML-MP 1.2版本的DOCTYPE。这是最新的版本。

没错，我知道前面提到“两行”，本来这确实是两行，只不过这里篇幅太小，第二行代码无法在一行上完全显示。

为什么我们要用这个旧东西？



我们只是改变了DOCTYPE声明。看起来这毫无意义。为什么不能直接使用HTML5呢？

我承认，表面看来，使用XHTML-MP确实看起来很让人沮丧……

它不支持我们在HTML中很喜欢使用的一些特性，而且现代手机已经越来越多地支持（或至少接受）HTML5标记。

……不过，修改了DOCTYPE之后确实发生了一些变化。

如果在Opera Mini 4.2模拟器中加载http://hf-mw.com/ch3/ex/2，你就会看到惊人的变化。显然使用XHTML-MP作为我们的DOCTYPE改变了浏览器的行为方式！

内容不再超越页面边界。相反，按钮和浮动元素都在页面宽度范围内。

原来是这样：Opera Mini把HTML5 DOCTYPE当作是一个“提示”，看到这个提示就会认为要表现得更像一个桌面浏览器。这说明，如果布局中有基于百分比的宽度，比如响应式布局，就可能不会按我们预期的那样工作。



嘿，看哟！现在内容全都很好地显示在屏幕上了。

Watch it!

我们的任务不会就此结束：不能只是修改DOCTYPE就认为大功告成。

我们已经把DOCTYPE修改为XHTML-MP，不过这并不表示标记确实是合法的XHTML-MP标记。我们还需要对代码做一些修改，让它成为真正的XHTML-MP。

XHTML-MP力求简单

尽管不会每分每秒都有惊喜出现，不过使用一个面向移动的DOCTYPE对我们确实有帮助，因为：

1. **可以确保很多老式移动浏览器也提供支持。**

   尽管大多数移动浏览器不会一遇到HTML5文档就“完蛋”，但这些老式浏览器确实有可能。而且HTML5中支持的一些特性在大多数老式手机上都不起作用。

2. **可以提醒我们潜在的移动圈套，让我们保持清醒。**

   如果一个特性在XHTML-MP中未得到支持，则肯定是有理由的。使用XHTML-MP并保持在其限定范围内，这样可以避免我们贸然进入危险领地。移动优先RWD会让我们重点关注约束、简洁性和最关键的任务，类似地，XHTML-MP会提供一个框架，在这个框架内可以创建一个能得到广泛支持的移动网站。

there are no Dumb Questions

问： 显然我失忆了。什么是DOCTYPE呢？

答： DOCTYPE（更正式的说法是文档类型声明）是放在SGML和XML文档最前面的一小段文本，通知客户（也就是浏览器）要根据哪个DTD执行这个文档。

问： 太多缩略语了。SGML和XML文档？DTD？这都是些什么？

答： HTML从SGML（标准通用标记语言，Standard Generalized Markup Language）发展而来，XHTML则是一种XML（所以与HTML相比，XHTML结构更严格），因此这二者都支持DOCTYPE。

文档类型声明（DTD）是一些形式化定义，指出哪些元素（标记、属性等）是允许的，另外在其描述的文档类型中这些元素可以放在哪里。每个HTML和XHTML规范都有自己的DTD。

正常情况下，如果为一个文档关联其指定DTD，提供显示的客户端就能根据这个DTD验证文档。但在实际的Web浏览器中，不会发生这种情况，Web浏览器从来不会真正根据DTD验证文档。

问： 那么DOCTYPE还有什么意义呢？为什么浏览器还一直需要DOCTYPE呢？

答： 浏览器会对DOCTYPE完成一种“嗅探”（与之前的user-agent嗅探没有太大不同）。尽管它们不会正式验证文档，但是会把DOCTYPE作为一个提示，了解将采用哪种“模式”，以及如何显示内容。

应该还记得前面的Opera Mini例子吧？把DOCTYPE从HTML5改为XHTML-MP（或者XHTML-Basic）会让布局模式改变。这就是DOCTYPE作为提示的一个很好的例子。

其他浏览器使用DOCTYPE来确定是以“标准”(standards)模式还是以“混杂”(quirks)模式显示内容，“混杂”模式是一种向后兼容、容忍度更高的模式。

对于HTML5，浏览器具体查看DTD的所有要求都会去除，正是因为这个原因，现在的声明相当简短，只是<!DOCTYPE html>（而不再包括一个DTD的URL）。



XHTML-MP闪亮登场

本周访谈：

为什么要用XHTML-MP?

坏脾气的移动Web开发人员（以下简称GMWD）：XHTML-MP，你可有一把年纪了。什么时候你能买下大农场养老，或者把它卖给HTML5?

XHTML-MP 1.2：我还没死呢！有些人以为我已经行将就木了，不过我可是个有用的老家伙。我能让那些老式移动浏览器继续发挥作用。你可以信赖我，我能帮你摆脱困境。

GMWD：你能帮我摆脱哪种困境?

XHTML-MP 1.2：想让我举几个例子？好吧。如果你在使用一个移动设备，可能设置一个锚(<a>)标记同时指定目标为_new或_blank，这有什么意义？很多移动浏览器不允许从链接打开一个新窗口。所以我不支持目标（target）。噢，还有框架（frame）我也不支持。它们实在是危险的东西。我不支持任何框架，甚至那些漂亮的iframe。

GMWD：不支持iframe？哼！

XHTML-MP 1.2：别忘了，我们的XHTML-MP家族（从我祖父XHTML-MP 1.0还是个孩子时算起）会为你提供访问键。

GMWD：访问键?

XHTML-MP 1.2：我们整个XHTML-MP家族都会在锚(<a>)标记上提供访问键（accesskey）属性。允许你指定数字0～9作为这些锚的快捷键。这对于有数字键盘的手机很有用，还有助于省去麻烦的滚动。

GMWD：这么说……XHTML-MP发明了访问键?

XHTML-MP 1.2：嗯，理论上讲，这是好几代传下来的，从WML到C-HTML再到我们。这确实是个传家宝。

GMWD：WML? C-HTML?

XHTML-MP 1.2：我还没死，不过这些家伙确实已经不在世上了。无线标记语言（Wireless Markup Language，WML）被逐步淘汰直到最后完全消失。它与HTML差别很大，我们XHTML-MP家族取代它成为了首选的移动标记语言。

C-HTML（精简HTML，Compact HTML)以前曾在日本大量使用，主要用于DoCoMo。有些人把它叫作iMode。想要表情符号吗？ C-HTML提供了丰富的表情符号。不过它还缺少很多东西，没有表格、没有CSS、没有图像。噢，还没有颜色。啊，真怀念从前的日子。



保证标记简洁。

我们非常关心浏览器会不会“完蛋”。与桌面浏览器相比，移动浏览器（特别是老式浏览器）对格式不正确的标记容错性较差。你需要特别用心地验证标记，不正确的标记可能导致移动浏览器崩溃，还可能更糟糕，甚至可能让手机重启。



Jim：这么说，Frank，你认为我们应该用XHTML-MP？

Frank：听起来这是最安全的办法。看来我们需要保证标记简单明了，才能让这些老式浏览器接受。

Jim：这是不是要求我们还要做些改变？

Frank：XHTML-MP DOCTYPE修正了Opera Mini中的浮动<div>，这一点很棒，不过我对于在这些老浏览器上实现浮动开始有些担心了。我知道我的想法有些老套，不过我想是不是可以使用表格来摆放这些仪表板信息呢！

Jim：老兄，HTML表格可是20世纪90年代的遗物啊。

Frank：通常我都像躲瘟疫一样躲着它，不过它们毕竟还是有效的。而且这些仪表板信息感觉确实很像表格数据。另外我实在不相信那些基于<div>的浮动分栏在我们要支持的所有移动浏览器中都能可靠地工作。

Jim：是啊，表格是有些老旧，不过我想大家都清楚它们起码能工作。另外……我们为这个模拟网站编写的那些CSS呢？

Frank：支持哪些HTML标记，另外我们可以依赖哪种CSS支持，这些问题实在让人崩溃。

我想还是一个问题一个问题地解决吧。我希望首先明确哪些HTML元素是合法的，这些手机能够支持。所以，现在先把所有CSS去掉，以后再加回来。

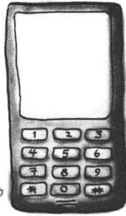


我们可以为#tools <div>中的链接指定访问键属性，这样能更快地访问到仪表板和其他链接。

不用大量滚动。

顺便说一句，还有滚动问题

并不是每个人都拥有触摸屏。很多手机上要在Web页面间导航，办法只有一个，就是利用一个按键或者光标滚动内容。在很多界面上，用户向下滚动页面时，每个可单击的项会突出显示，使用户有机会进入这些链接。链接和内容越多，滚动就越多。



访问键允许用户使用数字键作为快捷键！

……好在有访问键

anchor （<a>)标记上的accesskey属性允许我们为各个锚指定一个数字快捷键，来减少用户的滚动量。用户可以使用手机上的数字键快速访问链接。访问键的语法如下：

```
<a href="#dashboard" accesskey="1">Your Dashboard</a>
```

为链接指定的访问键数字不会自动显示，所以对于链接列表，使用有序列表(<ol>)而不是无序列表(<ul>)通常会很方便（假设列表中的第一项使用accesskey 1）。

最后一个问题

嗯，还有一件事忘了告诉你。也许这会让你大发脾气，想甩手不干。我们能理解。

> 很抱歉，XHTML-MP 1.2在2008年被XHTML-Basic所取代。

XHTML-Basic 1.1提供了XHTML-MP 1.2的所有内容，另外还增加了几个特性。锚标记又有了target属性（不过我们并不是建议你非得使用这个属性）。另外还有了一些新标记，比如<sup>和<sub>。关键是，你不会失去XHTML-MP 1.2已有的东西。

另外，XHTML-MP 1.2确实能在移动浏览器中工作（而且还不错）。

Relax

对于移动设备，有很多让人眼花缭乱的标记可以选择，不过本章后面将使用XHTML-Basic。

很抱歉前面讲了一堆已经被取代的东西，不过这是为了你好。移动Web标记领域很复杂，你应该对现状有些认识。下面我们不再闲聊了，接下来会使用XHTML-Basic。这并不太难，我们只需要明确不支持哪些标记，并避开这些标记就行了。另外目前甚至不用担心CSS样式或布局的问题。

试一试

1. **将文档转换为XHTML-Basic。**

   将当前的DOCTYPE标记替换为以下标记：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN"
"http://www.w3.org/TR/xhtml-basic/xhtml-basic11.dtd">
```

2. **现在先删除CSS。**

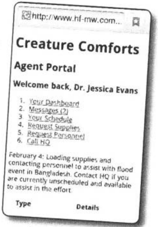
   删除CSS <link>标记，后面再处理CSS。

3. **将#tools <div>中的<ul>转换为<ol>。**

   这会在链接旁边显示数字，对下一个步骤有帮助。

4. **为#tools <div>中的<a>标记增加accesskey属性。**

   语法提示见第119页。



Android设备上的模拟网站，尽管简陋，但确实能工作。

5. **把基于<div>的浮动#dashboard <div>转换为一个表格。**

   将#dashboard <div>中的内容替换为一个表格。这有点麻烦，所以你可以使用第121页上现成的标记（或者可以在chapter3文件夹的extras目录中找到这个代码，具体在一个名为table.txt的文件中）。

一定要保留#dashboard <div>，只是替换它的内容。

6. **将修改后的文件保存为index.html。**

   将这个文件保存为index.html，现在我们完成了移动重定向的测试，所以使用index.html更为容易。

there are no Dumb Questions

问： 嗯，我知道我们现在要努力简化，不过，把DOCTYPE改为XHTML-Basic之后，115页上看到的模拟网站还能在Opera Mini 4.2中正常工作吗？

答： 可以！如果你很好奇，可以到http://hf-mw.com/ch3/ex/2a查看。

问： Opera Mini 4.2太老了，不是吗？到底还有多少人使用这个版本呢？

答： 必须承认，这确实不是一个时新的浏览器。不过，在移动Web的荒蛮世界中，这种老旧的浏览器确实存在。老式功能手机上（特别是新兴市场中）通常都有这种存在某些约束的浏览器，这就是一个很好的例子。

问： 不过，你不是刚刚说过智能手机的销量现在超过了这些老旧的功能手机吗？

答： 销量超过并不代表数量超过。现在仍然有大量老式的智能手机（按我们目前的标准来看，它们已经不算"智能"了）和功能手机。还需要相当长的时间它们才会逐渐消失。

成品表代码

```
<table>
  <thead>
  <tr>
    <th>Type</th>
    <th>Details</th>
  </tr>
  </thead>
  <tbody>
    <tr>
      <td><p>Message<br />Feb 3 8:54PM EST</p></td>
      <td><p><a href="#">Supply Request #493-C4 Approved</a><br />
    Hi, Jess, Good news! I wanted to let you know that we were successful in
tracking down those bottles...</p></td>
    </tr>
    <tr>
      <td><p>Message<br />Feb 3 1:47PM EST</p></td>
      <td><p><a href="#">Supply Request #493-C4 Received</a><br />
      This is an automated message to confirm that your recent Supply Request
has been received and is in process...</p></td>
    </tr>
    <tr>
      <td><p>Event Scheduled<br />Feb 3 8:22AM EST</p></td>
      <td><p><a href="#">Itinerary 39924 Approved LAX -> DAC</a><br />
      Your itinerary for event: "Bangladesh Flood Event" has been approved.
Your calendar has been updated...</p></td>
    </tr>
    <tr>
      <td><p>Personnel Event<br />Feb 2 9:23PM EST</p></td>
      <td><p><a href="#">Re: Personnel Confirmation 03/05 - 03/15</a><br />
          Jessica! Thanks so much for committing to this operation! I think
Dr. Madling is going too and...</p>
      </td>
    </tr>
    <tr>
      <td colspan="2"><div class="morelink"><p><a href="#">More &gt;&gt; </p>
</a></div></td>
    </tr>
  </tbody>
</table>
```

Exercise

哪些元素合法，可以放在哪里？使用我们学到的知识，并利用你的直觉，确定各个标准中哪些标记和属性是合法的。一个元素可能在多个标准中都得到支持！

	XHTML-MP 1.2	XHTML-Basic 1.1	HTML5
<font>	☐	☐	☐
<table>	☐	☐	☐
<a target="">	☐	☐	☐
<sup>	☐	☐	☐
<a accesskey="">	☐	☐	☐
<u>	☐	☐	☐
<tbody>	☐	☐	☐
<link>	☐	☐	☐
<strong>	☐	☐	☐
<video>	☐	☐	☐
<iframe>	☐	☐	☐

→ 答案见第126页。



我们承诺过访问键在大量手机上都可用，但你不要太当真。

实际上大多数主要桌面浏览器中也支持访问键。

访问键的实际使用

如果你想对移动标记测试访问键，则可以在你的桌面浏览器中查看index.html。使用以下按键组合测试访问键。

Chrome 或Safari: CTRL-OPT [accesskey]
FireFox: CRTL-[accesskey]

Mac

Chrome 或 Safari: ALT-[accesskey]
FireFox: ALT-SHIFT-[accesskey]
Internet Explorer: ALT-[accesskey]

Windows

Chrome: ALT-[accesskey]
FireFox: ALT-SHIFT-[accesskey]

Linux

某些操作系统快捷键或软件配置可能有更高的优先级，所以你的组合可能有所不同。

做一些验证

我们说过使用合法的标记很重要，还记得吧？下面不再只是说说而已，现在我们将在模拟网站中验证标记，确保它们确实是合法的标记。接下来进入非凡的W3C标记验证工具（W3C Markup Validator）网站检查我们的代码。

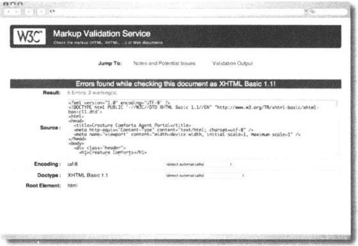


http://validator.w3.org

可以上传index.html文件（validate by File upload）或者复制粘贴其内容（validate by Direct Input）。

正常吗？可以继续吗？

哎呀！



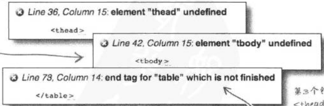
怎么回事？

向下滚动结果页面，查看有关错误的细节。那么到底是什么问题？

1 <thead>和<tbody>标记在XHTML-Basic（或XHTML-MP）中未得到支持。

XHTML-MP和XHTML-Basic中有些HTML模块未得到充分支持，这可能是其中最突出的了。表格可以使用，但是不能使用<thead>、<tbody>、<tfoot>、<col>或<colgroup>。

这3个错误都与移动模拟网站HTML中未得到支持的<thead>和<tbody>标记有关。



第3个错误是<tbody>和<thead>标记的一个副作用，如果删除这些标记，这个错误就会解决。

修正错误

2 **有一些嵌套不正确的标记。**

甚至资深专家也会犯类似这样的小录入错误。这也是使用验证工具的原因之一，即使你是顶尖高手也免不了犯错误。

index.html中。

```
<td colspan="2"><div class="morelink">
<p><a href="#">More &gt;&gt;</p></a></div>
</td>
```

```
Line 74, Column 80: end tag for "a" omitted, but OMITTAG NO was specified
…lspan="2"><div class="morelink"><p><a href="#">More &gt;&gt;</p></a></div></td>
```

```
Line 74, Column 84: end tag for element "a" which is not open
…lspan="2"><div class="morelink"><p><a href="#">More &gt;&gt;</p></a></div></td>
```

```
Line 74, Column 52: start tag was here
… colspan="2"><div class="morelink"><p><a href="#">More &gt;&gt;</p></a></div><…
```

实际上这不是一个错误，这是验证工具在为我们提供一个提示，告诉我们到哪里查找问题。

这两个错误都是由一个位置不当的标记造成的。

试一试

修正index.html文件中的标记问题。

1. 删除<thead>、</thead>、<tbody>和</tbody>标记。

2. 修正嵌套不正确的标记。把标记放在.morelink <div>的段落中。

3. 用W3C验证工具重新验证更新后的代码。



耶！绿色代表成功。

Exercise Solution

如何确定哪些标记在哪些标准中是合法的？

<font>和<u>都已经废弃，应该使用CSS!

	XHTML-MP 1.2	XHTML-Basic 1.1	HTML5
<font>	☐	☐	☐
<table>	☑	☑	☑
<a target="">	☐	☑	☑
<sup>	☐	☑	☑
<a accesskey="">	☑	☑	☑
<u>	☐	☐	☐
<tbody>	☐	☐	☑
<link>	☑	☑	☑
<strong>	☑	☑	☑
<video>	☐	☐	☑
<iframe>	☐	☐	☑



懂移动的CSS

CSS Mobile Profile 2.0 (CSS-MP)和XHTML-Basic（或XHTML-MP）在一起就像豌豆和胡萝卜形影不离。CSS-MP标准是为低到中端手机开发的。

在CSS Mobile Profile中能得到什么？嗯，可以得到CSS2所能提供的大多数特性，甚至还有一点点CSS3。不过不会是全部。

理论上听起来很简单

实现CSS Mobile Profile 2.0的浏览器原本应该支持必要的属性。不过事实却让人有些沮丧。对CSS的支持差异很大，这个责任通常会落到你头上，作为“天不怕地不怕”的开发人员，需要你全面彻底地测试，避开那些怪异的特殊情况。

另外，很多CSS属性和值在CSS-MP中认为是可选的，这说明浏览器制造商可以选择支持它们，也可能不支持。



最好的办法是：少说多做。

我们不打算抛出一个乏味的列表，列出CSS Mobile Profile中所有支持和不支持的属性。下面直接深入，明确如何调整现有的CSS做到CSS-MP兼容。

Long Exercise

系好安全带，做好准备，我们就要开始旅行了。如果一切顺利，等到旅行归来，我们就会看到XHTML-Basic和CSS Mobile Profile美满地结合在一起。我们会对index.html和styles.css进行编辑。

1 **把<link>标记放回到index.html中。**

我们又想使用CSS了，所以需要这个<link>标记。

```
<head>
  <title>Creature Comforts Agent Portal</title>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1,
maximum-scale=1" />
  <link rel="stylesheet" type="text/css" href="styles.css" />
</head>
```

index.html

2 **下面重构styles.css，让它成为合法的CSS移动简档（CSS Mobile Profile）。**

background-image和background-repeat属性在CSS-MP中是支持的，不过如果有一个包含背景图像的大页眉，这会浪费很多空间。下面去掉背景图像。

background-position属性的百分数和像素值在CSS-MP中并不支持，所以如果要保留背景图像必须进行修改。

不过我们不打算保留。设备支持存在不确定性。真糟糕，甚至对background-color的支持也不能保证。

```
* {
  padding: 0;
  margin: 0;
}
body {
  font-family: "Helvetica", "Arial", san-serif;
  font-size: 100%;
  width: 100%;
  background-color: #f3ffc2;
  background-image: url('cows.jpg');
  background-repeat: no-repeat;
  background-position: 50% 0px;
}
p {
  font-size: .95em;
  margin: 0.25em 0;
}
h1, h2, h3, h4, h5 {
  font-family: "Times New Roman", serif;
  margin: 0;
  color: #10508c;
  text-align: center;
}
```

（以上代码中 background-image、background-repeat、background-position 三行已被划掉。）

styles.css

这个CSS的所有内容到目前为止都是兼容的……继续前进！

styles.css

```
h3 {
  font-style: italic;
  font-weight: 100;
  font-size: 1.15em;
}
ol {
  width: 100%;
}
a {
  text-decoration: none;
  color: #096c9f;
}
.header {
 height: 150px;
}
#tools ol {
  list-style-type: none;
}
#tools ol li a {
  -webkit-border-radius: 5px;
  -moz-border-radius: 5px;
  border-radius: 5px;
  display: block;
  height: 1.1em;
  width:  94%;
  background-color: #fff;
  margin: 3%;
  border: 1px solid #ccc;
  text-align: center;
  padding: .6em 0;
}
.greeting {
  border: 1px dashed #10508c;
  border-width: 1px 0;
}
#dashboard {
  -webkit-border-radius: 5px;
  -moz-border-radius: 5px;
  border-radius: 5px;
  background-color: #fff;
  border: 1px solid #ccc;
  margin: 1em 3%;
  padding: .5em 0;
  width: 94%;
}
```

从我们最后一次使用这个CSS以来，已经把<ul>改为<ol>。这些规则需要更新来反映这一点。

这些属性本来用来创建圆角，它们在CSS-MP中是不合法的，大多数老式浏览器不支持这些属性。将它们从CSS中删除。

继续翻到下一页。

Long Exercise

3 再回到index.html：为表格元素增加一些类。

```
<tr class="even">
  <td class="event event_meta"><p><strong>Message</strong><br
/>
  Feb 3 8:54PM EST</p>
  </td>
  <td class="event"><p><a href="#"><strong>Supply Request #493-
C4 Approved</strong></a><br />
  Hi, Jess, Good news! I wanted to let you know that we were
successful in tracking down those bottles...</p></td>
</tr>
```

嘿，漂亮！还要把这些类应用到另外3行。

不过tr.even和tr.odd要交替应用。

index.html

4 现在再来看styles.css：调整样式，应用到表格标记。

我们把<div>改为表格来建立布局。这里希望去除那些不再有意义的属性，另外对其他属性做适当的修改。

```
#dashboard td.event {
  border: 1px dashed #ddd;
  border-width: 1px 0 0 0;
  margin: .5em 0;
  padding: 0.5em 0;
  width: 100%;
  clear: both;
  overflow: hidden;
}
#dashboard .odd {
  background-color: #fff;
}
#dashboard .even {
  background-color: #eee;
}
#dashboard td.event_meta {
  width: 30%;
  margin: 0 2%;
  float: left;
}
```

删除这3个属性。（width: 100%; clear: both; overflow: hidden; 已划掉）

这里不再有浮动。另外对宽度和边距做些调整。（margin: 0 2%; float: left; 已划掉）

styles.css

不再让内容浮动。另外，实现CSS-MP的浏览器只需要实现一个overflow值（即“auto”），所有其他属性都是可选的。CSS-MP中不能依赖于overflow。

5 **将这个CSS规则增加到styles.css……**

增加这个新CSS规则（可以放在styles.css文件中的任何地方）。

```
#dashboard table {
  border-collapse: collapse;
  width: 100%;
}
```

styles.css

6 **……并删除一些不再需要的规则。**

找到并去除以下样式规则：

删除所有这些规则！

```
#dashboard div
#dashboard .event_time
#dashboard .event_details
#dashboard .event_subject
#dashboard .event_summary
```

styles.css

7 **修正#status_message <div>。**

```
#status_message {
  background-color: #f8f1b2;
  padding: 0.25em;
  line-height: 1.3em;
  border: 1px solid #ccc;
  border-width: 1px 0;
  height: 70px;
  overflow: scroll;
}
```

最后，去掉这些属性。

styles.css



Watch it!

使用overflow属性要当心！

overflow: scroll在移动领域是完全禁止的。任何平台一般都不支持，在交互性方面这是很让人讨厌的一个特性。

实现CSS-MP的浏览器中，overflow属性的行为也变化多端，现在只要求它们支持一个值：auto。要尽可 能避免依赖于overflow。

8 **保存这些文件，大功告成了！**

嗯……少了点东西

来看看Opera Mini 4.2模拟器上当前这个模拟网站的截屏图。注意到什么了吗？我们丢掉了<ol>的列表序号，要知道这些数字指示了访问键。唉呀，真不妙。

模拟网站现在的样子，没有访问键。



那些数字去哪里了？

那些数字还在“那里”，只不过我们应用了CSS，使<li>元素看起来像按钮，占据了窗口的整个宽度。所以这些数字从可见的内容左边消失了。

正常情况下，要“把它们找回来”，我们得使用类似这样的东西：

```
#tools ol {
  list-style-position: inside;
}
```

list-style-position: inside会把数字放在<li>元素“里面”，并把它们显示在<li>的内容（链接）旁边（对于我们这种情况）。听起来好像不错，是吗？

遗憾的是，list-style-position在CSS-MP中并不支持。所以我们遇到一个难题。可能需要把这些链接转换为一个简单的链接列表，而不是现在这样像按钮一样。这样数字就会再次出现了。

再见，按钮

通过将#tools ol和#tools ol li的规则替换为以下规则，我们就能再次得到有序号的列表了：

```
#tools ol {
  margin: 0.5em 1.5em;
}
```

=



如果把现在的CSS换成新CSS……

又及，所有元素(*)的内边距和外边距都设置为0。所以我们要再增加一点外边距，让数字显示在可见的页面上。

…<ol>的序号就回来了。

不过，Creature Comforts查看这个布局修改时有异议……

按钮外观没有了！



Creature Comforts坚持按钮外观，还是希望改回来。

要用合法的CSS-MP做到这一点，唯一的办法就是把列表再转换回<ul>，把数字增加到内容本身，然后更新CSS样式规则。

哈，让步吧。这就是移动Web的方式！

试一试

修改styles.css文件中的CSS选择器，在index.html中将<ol>转换为<ul>。不要忘记还要在链接的文本内容中增加访问键数字！

styles.css

```
#tools ul {
  ...
}
#tools ul li a {
  ...
}
```

这里加上第129页上#tools ol的CSS。

这里增加之前#tools ol li使用的CSS。

在index.html中把<ol>再次改回为<ul>。

```
<div id="tools">
  <ul>
    <li><a href="#dashboard" accesskey="1">1. Your Dashboard</a></li>
    <li><a href="#" accesskey="2">2. Messages (2)</a></li>
    <li><a href="#" accesskey="3">3. Your Schedule</a></li>
    <li><a href="#" accesskey="4">4. Request Supplies</a></li>
    <li><a href="#" accesskey="5">5. Request Personnel</a></li>
    <li><a href="#" accesskey="6">6. Call HQ</a></li>
  </ul>
</div>
```

index.html

大获成功！



BULLET POINTS

- 这是一个浩大的世界，有上百亿的移动手机（这是真的）。它们并不都是前沿的智能手机，有时需要让你的网站或Web应用适用于那些老式手机。
- 存在一些实际情况：较早的系统、难对付的客户，或者一些特殊的项目，这使得建立一个完全独立、面向移动设备的网站很有必要。
- 要把移动信息流路由到一个移动网站，方法之一是使用服务器端移动设备检测和重定向。
- user-agent嗅探是检查进入的请求是否来自移动浏览器的一种常用技术。
- user-agent嗅探检查浏览器发送的user-agent首部（作为HTTP请求的一部分）。用户可以“欺骗”他们的用户代理（可能有意或无意）。这是这种方法的一个缺点。
- 有些老式移动浏览器（以及当前的低端移动设备）实现了不同的HTML和CSS标准。
- XHTML移动简档（XHTML Mobile Profile，XHTML-MP）是很多移动浏览器使用的一个针对移动设备的标准。它与XHTML类似，不过不完全支持XHTML中的所有特性。
- 类似的，CSS移动简档（CSS Mobile Profile，CSS-MP）是一个针对移动设备的CSS标准。
- XHTML-MP被XHTML-Basic 1.1取代，后者与前者基本相同，只是支持的元素更多一些。
- 为移动Web项目选择适当的DOCTYPE很重要，要注意保证你的标记合法，不正确的代码可能导致手机表现很差。

there are no Dumb Questions

问：可以用XHTML-Basic/XHTML-MP而不是HTML，另外可以使用CSS-Mobile Profile而不是全面的CSS2。那么有没有专门针对移动设备的移动版本的JavaScript呢？

答：是的。确实有一个针对移动设备的JavaScript，名为ECMAScript移动简档（ECMAScript Mobile Profile），不过不要太依赖它。我们处理过一些手机，在那些手机上一旦页面加载，JavaScript就无法改变页面上的任何内容，这说明使用JavaScript做的大多数工作都无法完成。所以，如果你面向老式手机，则最好不要太指望JavaScript。

问：无线CSS呢？

答：无线CSS（Wireless CSS）是一个与CSS-Mobile Profile非常类似的标准，不过支持的属性更少。它已经渐渐没落，我们看不出有什么理由使用它而不是CSS-MP。

问：如果使用user-agent嗅探来路由移动信息流，则可能会错误地标识一个用户的浏览器，这可能因为用户欺骗了她的用户代理，也可能因为你……弄错了，总之用户最后是不是会被重定向，进入移动版本的网站，而无法返回到桌面版本？

答：问得好！这很重要。在真实情况下，你可能想提供一个"逃生舱"，通常采用的形式是提供一个返回桌面网站的链接。

仅仅这个链接是不够的，你需要让重定向脚本知道用户希望访问这个桌面网站。可以通过设置一个cookie来做到，指示这个首选项，如果浏览器设置了这个cookie，就不要完成重定向。

问：如果只采用完全合法的XHTML-Basic或CSS-MP之类的标准，无法让我的网站做我想做的事情，那该怎么办？是不是天就塌下来了？是不是全世界的手机都会崩溃？

答：要在移动Web领域取得成功，这意味着要做出让步和妥协。编写合法的代码当然是我们追寻的目标，但是并不总是100%可行。不过在破坏规则之前先要了解规则，这绝对是一个很好的座右铭。

问：是不是每次想要支持大量移动浏览器时都必须使用XHTML-Basic？感觉它很受限啊。

答：这里我们必须借用经典的移动Web回答："要看具体情况。"对于Creature Comforts这种情况，需要支持大量老式的、功能不太完备的手机。我们的目标是提供一个服务器端数据服务，尽可能在更多的功能手机上很好地显示。

很多情况下使用HTML5也是合理的方法，甚至是在老式手机上。不过我们希望向你展示面对老式设备时需要当心的这些问题。

问：你在第127页上没有给出列表，不过我确实想看看CSS-MP中支持和不支持的属性的完整列表。

答：嘿，你走运了！CSS Mobile Profile 2.0规范相当简洁易懂，可以在这里找到这个规范：http://www.w3.org/TR/css-mobile

BRAIN POWER

嘿，开发人员！我们怎么支持那些想要桌面版本的用户呢？你能想到从移动网站增加一个指向桌面网站的链接需要些什么吗？另外，要编写和检查一个存储用户首选项的cookie，需要对redirect.php做哪些修改？

如果用户想逃离到桌面版本，把他们限制在这个缩减版的移动网站上可不好。

4 决定支持谁

我们要支持哪些设备？



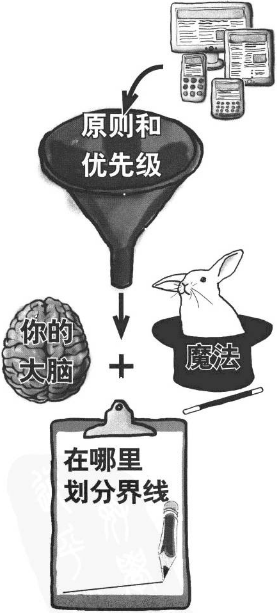
我们实在没有那么多时间在每一个设备上测试。

你必须划分出一个界线，确定能支持哪些设备。不过如何确定呢？有些人可能使用一些特殊的设备，而你无法在那些设备上测试，对于这种情况你该如何处理？完全不考虑他们吗？或者有没有可能采用一种合适的方式构建Web页面，使人们用你从未听说过的设备也能访问？这一章中，我们将结合项目的需求和用户的使用情况制作一种魔法剂，帮助我们确定支持哪些设备，以及对于不支持的设备该如何处理。

你怎么知道在哪里划分界线？

每个项目都是由一系列妥协和折中组成的。我们要考虑投资人的需求、用户的体验反馈，还要考虑到搜索引擎优化。如果有人不能体会和理解你的具体应用背景，则对他们来说，确定哪些方面会影响项目简直就像是有魔法一样。

确定你关注的设备也是类似的。需要有一定的原则和优等级，再动动脑筋，洒一点魔法灵感，就能得到一个决定成功的设备列表。



稍事休息



到目前为止，我们一直都在努力工作，凡事亲力亲为，不过就像拍一部电影一样，演员也有自己思考的时间，现在他可能目光游离，正在为一个棘手的问题苦思冥想。突然间，灵感来了，立刻又投入到他的角色中。

不同之处在于，现在你就是演员。你已经有了决定在哪里划分界线所需的全部工具。

不过，你可不想在键盘上找灵感，所以我们做个短暂的中场休息，先不考虑如何构建网站，而是讨论一些抽象的东西，把原则和优先级转换成你要支持的设备列表。休息之后，你的大脑就能更加专注，能更好地解决问题。

不要担心，我们很快就会回来，继续研究网站的构建。最好的导演都知道，他们要做的应该是为演员们提供一些指导，具体的表演要让演员自己完成。

我们需要划分的界线是什么？

要支持已有的所有设备，这是一个宏伟的目标。在每一个设备上完成测试一定会让你发疯的。你可能希望支持尽可能多的人和设备，不过为了理智一些，你必须能够回答3个问题。

1 **我们支持什么设备？**

你打算在什么设备或者哪些类型的设备上测试你的页面，确保它们能按预期的方式工作？

2 **我们不支持的设备会怎么样呢？**

怎样做才能确保尽管未在某些设备上测试，但这些设备仍能使用你的网站？

2 **我们不能支持的设备会怎么样呢？**

如果用户的设备不符合要求，该向他们提供什么消息？

BRAIN POWER

你能想出不支持和不能支持的设备有哪些不同？

不支持的设备与不能支持的设备

一旦知道要支持哪些设备，其他的都应该是不支持的设备。那么为什么还要区分不支持和不能支持的设备呢？



你可能没有明确支持一个设备，但这并不表示你构建的网站不能在这个设备上运行。如果你使用了良构的语义HTML标记和渐进增强来建立页面，那么除了你保证的那些浏览器和设备外，你的页面可能在更多的浏览器和设备上都能正确访问。

不过，有时一个浏览器可能太老，能力太弱，所以你根本无法支持这样一个浏览器。

例如，如果你在网上卖鞋子，则需要使用HTTPS来保证信用卡信息安全。不过有些老式手机不支持HTTPS。倘若出现这种情况，最合适的做法就是告诉这个人不能通过她的手机来购买鞋子。



不要太刻薄

如果你不支持某个人的浏览器，则要好言好语地指出。他可能没有最新的手机，但是完全可以用另外一个更好的浏览器。

另外，谁都不愿意被人说他的手机是个破烂货。要亲切地对待你的顾客。

Watch it!

不要排除不支持的浏览器。

你可能无法验证你的网页能够在一个浏览器中运行，不过这并不表示你要排除它，要求人们不要使用这个浏览器。如果你使用语义标记构建网站，则它将在很多浏览器中都能工作，包括你没有测试以及对与项目不重要的那些浏览器。

如果手机太老，连HTTPS都没有，不能支持这些手机我可以理解，不过，我觉得，如果首先发布一个基本HTML文档，然后采用渐进增强用CSS和JavaScript来逐步增强，那么它在任何地方都应该能 工作吧？



你说得对。在很多情况下，渐进增强意味着你可以支持数以百计的浏览器。不过不同浏览器上的效果可能不同。

从基本HTML起步，渐进增强这个文档，大多数Web页面应该都是从这个起点逐步发展的。有高级特性的浏览器可以得到类似圆角矩形和梯度之类的增强效果。

不过，即使是网站或应用的最低需求，基本HTML也可能无法提供。例如，一个视频游戏可能需要WebGL（这是一个图像库）来实现游戏场景。如果浏览器不支持WebGL，那么根本没有内容可以显示，也就是说，也许到几年这后，从WebGL渐进增强到WebGL 2时这才有可能吧！

WebGL2不存在，不过以后会有的，不是吗？

经验告诉我们，在打破规则之前你首先要知道规则是什么。一般都应当从基本HTML开始，然后使用渐进增强，除非你有充分的理由证明这样做不可行。

即使这种做法确实无法达到要求，也可以提供一个基本HTML文档，指出“抱歉，这个游戏需要WebGL支持”之类的消息，这也是一个不错的起点，可以在此基础上构建游戏。

关于你的项目问几个问题

听上去就像是在教你1加1等于2，不过要明确如何划分界线，第一步应当是仔细考虑你的项目。用户是谁？哪个功能是核心？哪些特性是可选的？最佳体验是什么？



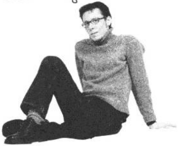
我们的Funny Cat Video网站在移动设备上表现很好，不过要求手机支持视频才行。哪些手机支持视频，有比较好的数据计划或WiFi能力呢？



我们准备把市场拓展到日本和韩国。他们和我们用一样的手机吗？我听说日本有很多别处没有的手机。



我们的最新游戏Enraged Avians有大量图像。我们已经知道浏览器需要支持WebGL，不过现在我们想知道如果手机支持WebGL，则它们会不会太慢。



PDG

移动魔法剂的配方

询问关于项目的问题时，就是在开始调查哪些因素可以帮助你做出决定，确定哪些是你需要关心的设备。你的用户群体是什么，他们可能使用什么设备？哪些是必须有的特性，而哪些是可有可无的特性？

这里就引入了魔法。把这些因素混合在一起可以得到一瓶魔法剂，告诉你需要支持哪些设备。



利用工具和数据

用户使用情况如何？可以深入研究Web分析，查看当前用户在使用哪些设备。要拓展到新的地区？可以抽取关于那些国家顶级手机的数据，比如查看新加坡最酷的手机是什么。



如果有一个橱柜放这些工具和数据，那么你的橱柜还空着吗？没关系，你很幸运。可以参考关于移动统计的一个指南，内容很详实，地址在：http://bit.ly/m-stats。

Exercise

该戴上你的魔法帽施展些魔法了。对于以下各个案例研究，根据这个应用要完成的工作和目标用户群体建立一个需求列表。

这个列表应当包括必须有的需求和可有可无的特性。

示例

案例研究

Mo Better Museums正在构建一个应用，允许参观博物馆的人把他们的手机相机对准某件艺术作品，可以看到一个特别说明，其中包含关于这件作品的更多信息。它能根据手机的位置指出正在参观哪件艺术作品。博物馆的手机信号接收情况可能不太好，时断时续。

需求：

- 必须有的特性：支持JavaScript。
- 必须有的特性：访问相机。
- 必须有的特性：访问位置。
- 可有可无的特性：支持离线模式。

这就是我们根据案例研究中的信息建立的需求列表。

案例研究

一个政治人物希望在最近的一次选举中胜出。她想为她的志愿者提供一个应用，可以在他们挨家挨户说服人们投票时使用。这个应用会提供要联系的投票人的地址，如果可能还可以提供一个地图，另外会给出志愿者要问的问题。志愿者很紧缺，所以她希望这个应用能支持他们可能使用的任何一款手机。

需求：

案例研究

Bowling Boxers是游戏Dogs Playing Poker的视频游戏版本。你要控制一个拳师犬，以著名的C. M. Coolidge作品中的狗为原型，它正在一场激烈的小巷滚木球赛中争抢。显示这些角色和疯狂的动作会让很多手机达到最大负荷。

需求：

案例研究

Global Corp注意到产品在印度的销量有所增长。销售部主管认为印度可能是Global下一个最大的市场。他希望提供Global网站的一个本地化版本，可以在印度最流行的手机上运行。他希望顾客能够尽可能容易地通过他们的手机下订单。

需求：

BRAIN POWER

哪些需求是必不可少的？另外哪些可以通过渐进增强来支持？

Exercise Solution

如果有人告诉你某个问题没有正确答案，你是不是很气恼？嗯，对此很抱歉。谈到需求，确实没有正确答案。

气恼如果你对这些案例研究的答案与我们不同也不用着急。根据你的经验，你可能会提出一个我们忽略的需求。相信你自己的经验和直觉。

EXAMPLE

案例研究

Mo Better Museums正在构建一个应用，允许参观博物馆的人把他们的手机相机对准某件艺术作品，可以看到一个特别说明，其中包含关于这件作品的更多信息。它能根据手机的位置指出正在参观哪件艺术作品。博物馆的手机信号接收情况可能不太好，时断时续。

需求：

- 必须有的特性：支持JavaScript。
- 必须有的特性：访问相机。
- 必须有的特性：访问位置。
- 可有可无的特性：支持离线模式。

相机访问是一个新的W3C API，到目前为止，只有最新版本的Android才提供部分支持。

案例研究

一个政治人物希望在最近的一次选举中胜出。她想为她的志愿者提供一个应用，可以在他们挨家挨户说服人们投票时使用。这个应用会提供要联系的投票人的地址，如果可能还可以提供一个地图，则另外会给出志愿者要问的问题。志愿者很紧缺，所以她希望这个应用能支持他们可能使用的任何一款手机。

需求：

- 必须有的特性：互联网连接
- 可有可无的特性：能够显示地图图像
- 可有可无的特性：提供交互式地图的JavaScript

太好了！这是实现渐进增强的机会！

案例研究

Bowling Boxers是游戏Dogs Playing Poker的视频游戏版本。你要控制一个拳师犬，以著名的C. M. Coolidge作品中的狗为原型，它正在一场激烈的小巷滚木球赛中争抢。显示这些角色和疯狂的动作会让很多手机达到最大负荷。

需求：

- 必须有的特性：支持JavaScript。
- 必须有的特性：支持WebGL。
- 必须有的特性：快速CPU和GPU，否则游戏性能会受影响。

案例研究

Global Corp注意到产品在印度的销量有所增长。销售部主管认为印度可能是Global下一个最大的市场。他希望提供Global网站的一个本地化版本，可以在印度最流行的手机上运行。他希望顾客能够尽可能容易地通过他们的手机下订单。

需求：

- 必须有的特性：为老式手机支持基本标记。
- 必须有的特性：为实现安全结账支持HTTPS。
- 必须有的特性：支持cookie实现登录。

Relax

交流可以帮助你发现额外的需求。

如果分析大多数移动网站，你会发现建立网站的人并不清楚他们需要支持哪些设备。他们只是尽可能做出最正确的猜想，然后在此基础上迭代调整，直到最后完善网站。

我怎么知道顾客有合适的设备呢？

既然已经有了一个设备需求列表，怎么知道某个手机满足我的要求呢？



问得好。主要有两个方法：服务器端设备检测和客户端特性检测。我们会在下一章介绍设备检测数据库。

第2章讨论渐进增强时我们介绍过特性检测。

看来你已经想结束中场休息，打算继续构建网站了。没错，我们说过不会休息太长时间。

现在让我们再整装待发，投身到疯狂的设备数据库世界中，来见识见识到处都能看到并影响Web开发人员的一大祸害：user-agent串。

别担心。它没有听上去那么糟糕。读完下一章，你就能很好地控制这些可能带来麻烦的user-agent串。

BULLET POINTS

- 每个项目都会对它支持的设备划分一个界线。
- 了解在哪里划分界线需要结合经验和研究，还需要有一点直觉。
- 不支持和不能支持的设备是有区别的，不能支持的设备是因为它们缺少一些关键的特性，使它们无法使用你的网站。
- 分析你的项目需求和目标用户群体，可以帮助你确定在哪里划分界线。
- 应当把渐进增强作为默认方法。这样一来，除了你公布可以支持的那些设备，你的网站还能在更多的设备上正常工作。
- 除非你很确定地知道不能支持某个设备，否则不要排除这种设备。新的浏览器一直在不断涌现。要给它们一个机会证明自己符合要求。
- 不要被移动的复杂性吓倒。你已经掌握了所需的所有工具，利用它们完全可以知道要在哪里划分界线。要相信你的经验和直觉。

5 设备数据库和类

分组行动



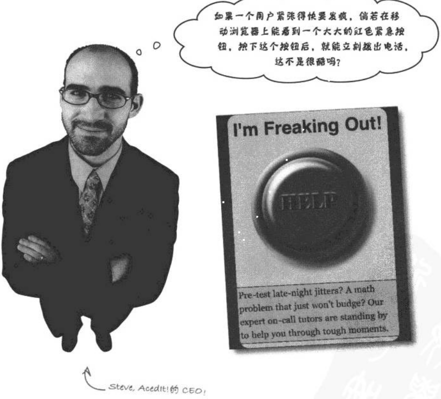
尽管能确定我们支持的设备，但有些烦人的小问题还不能解决。

如何了解用户移动浏览器足够的信息，在向他们发布内容之前能够知道这些浏览器确实符合要求？如何避免只为那些最低档次的设备构建内容（往往很简陋）？另外如何合理组织所有内容，以免我们焦头烂额？这一章中，我们将进入设备能力的领地，学习如何用设备数据库访问设备能力，最后还会介绍如何将它们归组为设备类，使我们能安然应对。

学生有麻烦，提供紧急按钮

AcedIt! Test Prep专门致力于为用户提供高级客户服务。其目标是针对每一个学生，这些学生在极其刻苦地学习，为他们的职业生涯做准备，他们精神高度紧张，可能要参加标准化结业考试，或者要考取某些证书，AcedIt! 正着力让他们能够与作为学科领域专家的导师建立一对一的联系。

有时学生可能慌乱得手足无措。正是由于这个原因，AcedIt!希望在网站上创建一个页面，专门帮助那些精神紧张的客户，调节他们的情绪。这个页面名为I'm Freaking Out!（我要发疯了），目标很简单，让客户随时与某个待命的导师联系。



这个按钮只针对移动手机

桌面浏览器不能拨电话，所以在桌面浏览器上看不到这个巨大的紧急按钮。Steve只希望这个按钮出现在移动手机上。



不过怎么知道用户在用移动手机呢？



Frank：也许可以根据浏览器的窗口宽度，并利用CSS媒体查询来判断吧。

Jill：在这里不行，我反对。一个窗口窄并不表示浏览器在真正的移动手机上。

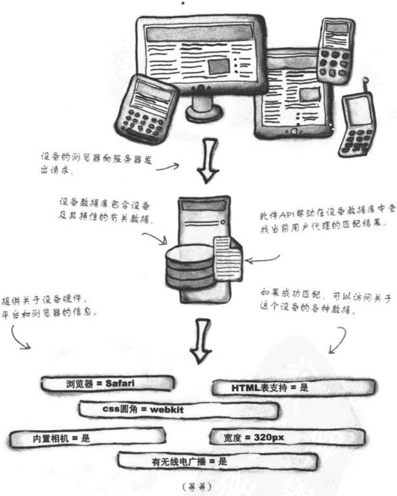
Frank：有道理，我想如果我们使用一个基于宽度的媒体查询，最后在窄窗口浏览器以及平板电脑等设备上可能都会出现这个紧急按钮。并不是所有移动设备都是手机。不过还有什么选择呢？

窄窗口并不代表手机！

Jill：理想情况下，我们要找到某种方法得到用户浏览器和设备的有关信息，这些信息要足够特定，能提示这是不是一个手机，或者提示我们它只是一个分辨率低或屏幕窄的设备。

移动设备数据源可以帮忙

移动设备数据库包含关于移动设备的浏览器、平台和硬件特性的详细信息。如果提供一个唯一标识的键（通常是一个user-agent串），通过查询这个数据库，我们可以得到有关这个设备及其浏览器的大量属性。



有了这些数据，就可以根据设备的特性来调整内容。

来认识认识WURFL

WURFL（无线通用资源文件，Wireless Universal Resource FiLe，通常读作“wuhr-full”）是一个开源的设备数据库，包含大量关于移动设备及其浏览器的特定能力的信息。

最近，长期以来一直维护WURFL项目的一些人新成立了一家公司，名为ScientiaMobile，专门为WURFL提供商业支持。

WURFL会跟踪移动设备特性的详细信息，从铃声格式到物理屏幕大小都会记录。

作为Web开发人员，我们并不是对WURFL的所有能力都感兴趣，不过确实可以在其中找到一些非常有用的宝贝。

Watch it!

设备不是平台，也不是浏览器。

目前还没有一个合适的词来表达“一个设备的硬件特性以及其平台、操作系统和浏览器”。确实应该有这样一个词，能清楚地表示出上述含义。WURFL中的“设备”概念就是这三者的组合，包含了硬件、操作系统和浏览器的有关信息。

只看“设备”这个术语很难清楚地了解它的真正含义。当然对此我们也有责任，所以要记住，尽管我们只是使用了很简单的一个词“设备”，但它往往同时隐含了浏览器、硬件和操作系统。

Exercise

自己动手试试看！利用ScientiaMobile的资源管理器，你可以在Web浏览器中与Tera-WURFL数据交互。

1. 在一个移动浏览器中访问http://www.tera-wurfl.com/explore

2. 访问“See my capabilities”（查看我的能力）部分，浏览它为你的移动设备和浏览器返回的能力列表。

这个资源管理器要在移动浏览器上使用。如果用桌面浏览器访问，会为你选择一个随机的移动设备，使你能看到某个设备的能力。不过这只是为了让你看到有结果，并没有实际意义（也就是说，你看到的并不是你的桌面浏览器的能力）。

还可以在一个桌面浏览器中访问这个资源管理器，输入任何user-agent串来查询相应设备的能力。对于一个已知的用户代理，要查询相应设备的能力，这是一种快速简便的方法。



在iPhone 4上，资源管理器中列出的部分能力。

WURFL及其能力



WURFL采用能力（capabilities）集合的方式提供移动设备、平台及其浏览器的有关信息。可以使用这个信息相应地调整你的Web内容。

目前WURFL跟踪记录500多个能力，分为20多个组，比如css（这个浏览器能支持圆角吗？背景图像呢），再比如playback （这个设备能播放哪些类型的媒体）。

设备数据在一个很大的XML文件中维护，这个文件会定期更新。最新版本可以在这个项目的Sourceforge页面找到。

毫无疑问，它的URL是wurfl.sourceforge.net。

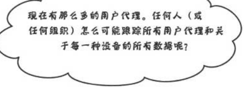
不只一种WURFL

ScientiaMobile的资源管理器建立在WURFL的一个特定实现基础之上（即WURFL的数据库版本，WURFL Database Edition，也称为Tera-WURFL），它将WURFL设备数据保存在一个数据库中，而不是放在平面XML文件中。建立Tera-WURFL的目的是作为一个Web服务或PHP库，移动网站可以查询这个服务得到user-agent串相应设备的信息。

有很多不同的WURFL API和实现，不过它们都有一个共同点：都是一个庞大的WURFL XML文件，包含所有移动设备数据。完整的WURFL实际上是一个组合，包括WURFL数据文件（XML）、 存储数据的方式以及与之交互的API。

我们将使用PHP API

目前针对多种主要编程语言有多个WURFL API。我们将使用PHP API和基于文件的数据。





WURFL有很多特有的特点（其中之一就是它并不跟踪每一个用户代理），使它能很好地工作。

WURFL的开发群体长期以来一直都在努力维护这个数据源，这说明并不只是某个可怜的家伙窝在小屋子里痛苦挣扎，试图跟踪市场上每一个新设备和浏览器，这不仅吃力不讨好，而且几乎是不可能的。

各个WURFL API的算法完成用户代理匹配时很聪明。不仅如此，WURFL已经存在一段时间，已经有大量可信并得到证实的记录数据，而且很多主要的Web用户都在使用（同时也在为它作贡献），听说过Facebook吗？

当然，如果WURFL不能满足你的需求，还有另外一些设备数据库可以使用，如DeviceAtlas、DetectRight和MobileAware。你的项目应该选择哪一个设备数据库，这取决于你的需求、实际预算，以及许可证需求。

there are no Dumb Questions

问：什么是ScientiaMobile？

答： ScientiaMobile是2011年由Steve Kamerman、Luca Passani和Krishna Guda成立的一家公司。Luca从2001年与Andrea Trasatti开始启动WURFL以来，一直是开源WURFL项目的维护者之一。Steve创建了Tera-WURFL，它已经成为现在的WURFL Database Edition。

问：WURFL免费吗？我要付钱吗？

答： 这要看情况。WURFL理论上讲是开源的。WURFL API在Affero通用公共许可证第3版（Affero General Public License v3，AGPL）下可用，这也是一个开源许可证。WURFL XML数据库则有一个受限的许可证，只允许结合WURFL API使用。

所以，如果你能遵循AGPL限制，则不必付任何费用。不过，AGPL比GPL更有"侵略性"。采用AGPL，倘若在服务器上运行一个软件，就会被统计为一个发布，这就必须遵循相关规定，要求你对所有衍生工作开源。

如果AGPL对你不适用（在很多情况下可能都不适用），ScientiaMobile会向你出售一个商业许可证。

问：如果我要集成WURFL和（这里插入你最喜欢的某个开源方案），并在其基础上构建了一个网站，我必须将整个网站开源吗？

答： 答案很简单，可以购买一个商业许可证，或者与你的律师谈谈。ScientiaMobile.com还提供了一个许可证FAQ板块。

问：有没有其他产品也能提供WURFL的功能？

答： 有。WURFL是唯一提供开源许可证的设备数据库，不过确实还有很多商业数据库。Device Atlas可能是最知名的一个产品了。其他产品还包括Mobile Aware和Detect Right。

问：除了PHP API以外，WURFL还有哪些API？

答： ScientiaMobile除了PHP API外，还提供了Java、.NET和Database API。

问：WURFL数据如何更新？更确切的讲，我怎样才能在WURFL更新时得到新数据？

答： ScientiaMobile一直在关注新设备。有时制造商会直接提供信息。其他情况下，都是使用WURFL的人提交信息。设备数据库开发商所做的很大一部分工作就是要验证别人提供的关于新设备的信息。

问：谁来决定WURFL跟踪哪些能力？

答： ScientiaMobile会根据群体提出的建议来选择跟踪哪些能力。

问：你说的这个WURFL群体在哪里？

答： 这个群体是一个邮件列表。WURFL很早以前就有了，所以这个列表名叫WML编程列表（很古老的名字）。另外ScientiaMobile.com上的支持论坛也会有相关讨论。

这个wml编程列表可以在http://tech.groups.yahoo.com/group/wmlprogramming/找到。

WURFL：聪明的API代码

WURFL API尝试将输入的一个用户代理与一个已知设备匹配时，如果数据中没有指出具体的用户代理，它也不会不知所措。由于跟踪已有的每一个用户代理是根本不可能的，所以API匹配函数在没有准确识别出用户代理时会采取一些聪明的小技巧。

设备家族树

WURFL API分析一个给定的user-agent串时，会评判一系列候选情况（这些候选设备的通用性依次增加），目标是至少把这个设备归类到适当的相关设备家族中。

可以把WURFL的数据看作是一种设备树，树干代表一个通用浏览器，各个分支、小枝杈和叶子代表更特定的设备或设备组。API尝试尽可能走得更远，达到具体的叶子，不过如果找不到，则落到树中的其他部分。

另外，一个给定设备的WURFL数据只定义了该设备（或设备组）与其父设备或祖先设备不同的能力。这样一来，WURFL XML数据文件可以相当小（到2011年年中，WURFL数据文件的大小略小于10 MB），尽管它包含的信息量确实相当大。

Geek Bits

WURFL API使用一个算法组合来避免一些奇怪的user-agent小特例无法处理。

WURFL API查找匹配时采用的一种方法是，使用针对一个给定浏览器或设备家族优化的匹配函数（matcher）。最新的PHP API有二十多个处理程序来处理用户代理的分析。

这个API首先确定对于给定的用户代理哪个处理程序最合适。例如，如果用户代理串为“BlackBerry”，就可以交给BlackBerryHandler处理。

每个处理程序都非常了解其设备家族中用户代理的种种重要差异，这样一来，WURFL数据维护者需要跟踪的特定用户代理数就可以大大减少。

PHP WURFL API结合使用处理程序和RIS匹配（即串内归约，reduction-in-string，也就是去除user-agent串中未识别的部分，直到得到一个确实能识别的串），来得到其最终结果。



该在你的计算机上安装WURFL了

查看附录iii，了解安装的详细信息。

我们也可以建立一个资源管理页面

一旦安装了WURFL PHP API，只需要很少的代码就能构建一个资源管理页面，而且与ScientiaMobile资源管理器页面相差无几。实际上，现在我们就要做这个工作。



自己动手建立一个资源管理页面，这会让你有机会具体使用WURFL API，另外可以使用你自己的数据文件中的数据。

这里所说的"你的数据"，是指你安装WURFL时下载的WURFL XML文件中的当前数据。

通过实践来学习，这才是上策。这里我们会做一些基础工作，可以在其他任务中重用（完成更多实际工作）。

另外，ScientiaMobile资源管理器页面是由WURFL数据库版本驱动的。数据的存储和匹配会与我们将要使用的XML文件-PHP API组合稍有区别。关键是，数据在很少（不过有时至关重要）的几方面稍有不同。

另外，通过建立你自己的资源管理页面，你很清楚在从你的数据文件中获取数据，版本是确定的（不会更老或更新）。

建立资源管理页面的步骤

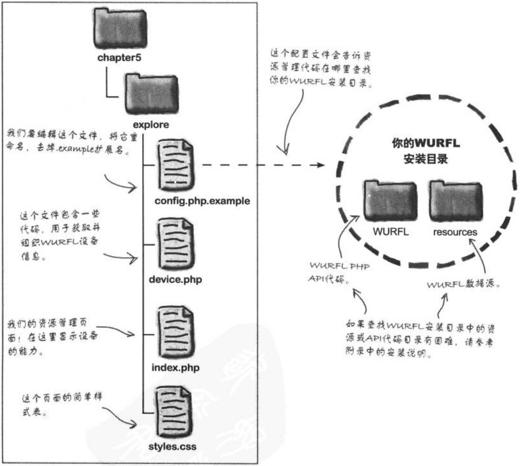
- [ ] 为WURFL建立工作环境、文件和配置。
- [ ] 写一些（样板）PHP代码初始化一些WURFL对象，从而能访问当前浏览器和设备的能力信息。
- [ ] 组织能力数据。建立一个页面，用一个HTML表格输出这些数据。

资源管理页面：建立我们的环境

第一步是组织目录结构并创建一个配置文件。可以在chapter5文件夹的explore子文件夹中找到一个配置文件作为起点。

现在你应该已经安装并运行了WURFL PHP API。要创建一个配置文件（嗯，当然是一个能正常工作的配置文件），必须知道安装的位置。

资源管理页面的目录结构



Long Exercise

你已经安装了WURFL（希望如此），不过现在我们需要适当地进行配置，让它能够在我们建立的web页面中工作。为此，我们要创建一个配置文件。

在一个文本编辑器中打开config.php.example。如下所示，编辑其中的路径，然后把这个文件保存为config.php，放在explore文件夹中（删除.example扩展名）。

```
/* WURFL_DIR needs to point to the install directory for WURFL */
define("WURFL_DIR", '/path/to/WURFL/');
/* RESOURCES_DIR needs to point to the resources dir you want to use. */
define("RESOURCES_DIR", '/path/to/WURFL/resources/');
```



config.php.~~example~~

请把它改掉！你要编辑这些路径，指向你自己的WURFL API代码和资源目录。

编写一次，无限重用。本章余下的例子中还会重用这个配置文件。

不要忘记去掉.example扩展名。

- [x] 建立WURFL工作环境、文件和配置。
- [ ] 写一些（样板）PHP代码初始化一些WURFL对象，从而能访问当前浏览器和设备的能力信息。
- [ ] 组织能力数据。建立一个页面，用一个HTML表格输出这些数据。

如果需要WURFL安装路径的相关帮助，请参见附录iii。

现在我们可以编写一些代码初始化WURFL对象，来访问设备的能力……

好了！我们已经有了配置文件。现在要创建一些代码初始化WURFL对象，有了WURFL对象我们就能深入了解设备的能力了。在文本编辑器中打开device.php。开始时这个文件很空，没有多少内容，所以下面插入一些关于WURFL的代码！

device.php的工作有两方面：使用当前请求的User-Agent首部（即用户的浏览器）初始化WURFL对象，以及组织设备的能力数据，以便在页面上显示。

这里包含我们新建的配置文件。

```
<?php
require_once('config.php');
$user_agent = $_SERVER['HTTP_USER_AGENT'];

$wurflConfig = new WURFL_Configuration_XmlConfig($wurflConfigFile);
$wurflManagerFactory = new WURFL_WURFLManagerFactory($wurflConfig);
$wurflManager = $wurflManagerFactory->create();
$device = $wurflManager->getDeviceForUserAgent($user_agent);
```

device.php



这几行代码会实例化几个WURFL对象，并为我们填充一个设备对象。

这个代码块中大多数都是样板代码，本章后面使用WURFL时将根据需要剪切粘贴这些代码。如果你很熟悉PHP，这些代码对你来说就是小菜一碟，不过，如果不熟悉也不用太担心。

继续翻到下一页。

Long Exercise

下面来仔细分析163页上样板代码块中的最后一行代码。在这里我们告诉WURFL如何建立一个设备对象。我们使用了$user_agent，现在它包含请求客户user-agent首部的值。

也就是说，我们要指示WURFL根据当前用户浏览器的user-agent串，尝试在其数据文件中找到相应的匹配。如果成功匹配，我们就有了一个设备对象，其中包含浏览器和设备的能力信息。

```
$device = $wurflManager->getDeviceForUserAgent($user_agent);
```

如果执行到这里，而且一切正常，说明我们可以使用设备能力信息了。

WURFL_WURFLManager类中的getDeviceForUserAgent方法取一个user-agent串，并（由WURFL数据）返回匹配的设备，设备将作为一个WURFL_CustomDevice对象返回。

可以这样翻译这句话：我们输入一个user-agent串，它返还我们一个设备对象，其中填充了设备的各个能力值。

- [x] 建立WURFL工作环境、文件和配置。
- [x] 写一些（样板）PHP代码初始化一些WURFL对象，从而能访问当前浏览器和设备的能力信息。
- [ ] 组织能力数据。建立一个页面，用一个HTML表格输出这些数据。

既然有了一个已填充的WURFL设备对象，下面组织这个设备的能力信息，以便在一个HTML表格中显示。

device.php的第二个任务是组织设备的能力信息，以便以后显示。我们将在文件中增加一个代码块，按WURFL组来组织这些能力。

在device.php中加入以下代码，并保存。现在我们已经完成了device.php文件。

getListOfGroups()方法会返回WURFL组名构成的一个数组……

……getCapabilitiesNamesForGroup()获取这个组中所有能力的名字！

```
$device = $wurflManager->getDeviceForUserAgent($user_agent);
if ($device) {
  $groups = $wurflManager->getListOfGroups();
  $grouped_capabilities = array();
  foreach($groups as $a_group) {
    $grouped_capabilities[$a_group] = array();
    $capabilities = $wurflManager->getCapabilitiesNameForGroup($a_group);
    foreach ($capabilities as $cap) {
      $grouped_capabilities[$a_group][$cap] = $device->getCapability($cap);
    }
  }
}
```

device.php

这个代码块分组组织当前设备和浏览器的能力。

我们通过这个方法访问各个能力的值。稍后还会更详细地介绍。

如果你不能完全理解这个PHP在做什么，也没有关系。只需要把它复制到device.php文件中就可以了。

还要翻到下一页。

Long Exercise

现在我们已经完成了device.php。下面要编辑index.php并增加一些代码。

- [x] 建立WURFL工作环境、文件和配置。
- [x] 写一些（样板）PHP代码初始化一些WURFL对象，从而能访问当前浏览器和设备的能力信息。
- [ ] 组织能力数据。建立一个页面，用一个HTML表格输出这些数据。

这部分已经完成！

该做这个工作了。

成品 PHP/HTML

迭代处理收集到的WURFL能力数据（按组组织）。

对于每一组，用该组中各个能力的名和值生成一个<dl>。

下面显示用户代理和设备的id属性（该浏览器或设备的WURFL ID）。

```
<div id="devicedata">
  <h2>Device Data</h2>
  <p>Device data for <?php print $user_agent; ?></p>
  <p><strong>WURFL Device ID: <?php print $device->id; ?></strong></p>
  <?php foreach($grouped_capabilities as $group_name => $my_caps): ?>
    <h3 class="group"><?php print $group_name; ?></h3>
    <dl>
      <?php foreach($my_caps as $cap_name => $cap): ?>
        <dt><?php print $cap_name; ?></dt>
        <dd><?php print ($cap) ? $cap : '[no value]'; ?></dd>
      <?php endforeach; ?>
      </dl>
  <?php endforeach; ?>
</div>
```

如果当前能力没有具体值，则显示串"no value"（无值）。

在各个<dt>中显示能力名，在各个<dd>中显示能力值。

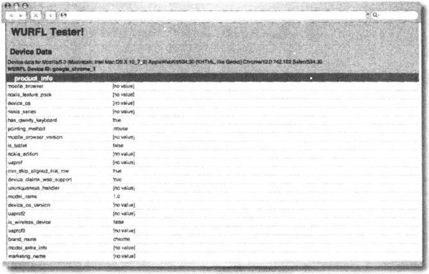
OK，继续前进！保存index.php，并在你的浏览器中查看这个页面。

index.php

不错的起点！

现在可以在桌面浏览器中看到所有能力，这些能力分组组织。

- ☑ 建立WURFL工作环境、文件和配置。
- ☑ 写一些（样板）PHP代码初始化一些WURFL对象，从而能访问当前浏览器和设备的能力信息。
- ☑ 组织能力数据。建立一个页面，用一个HTML表格输出这些数据。



不过还可以更好

我们不希望只显示当前浏览器的能力（尤其是如果正在使用一个桌面浏览器，这确实没什么意思），下面让资源管理页面可以显示任何用户代理的WURFL数据。为此，可以在index.php中加入一个简短的HTML表单，并对device.php稍做一点改动。

完成一个快速的双连击，改进我们的资源管理页面

做两个简单的修改，我们的资源管理页面会变得更有用。

$_SERVER['PHP_SELF']是当前执行的脚本的简写。也就是说，这个表单将提交到当前页面。

```
<div id="testform">
<form method="post" action="<?php print $_SERVER['PHP_SELF']; ?>"
id="useragentform">
  <p>Test this user agent string:</p>
  <input type="text" name="useragent" id="useragent_field"
    value="<?php print $user_agent; ?>" /><br />
  <input type="submit" name="submit" value="Test User Agent" id="submit" />
</form>
</div>
```

增加一个小表单，允许输入任何user-agent串。

index.php

```
require_once('config.php');
$user_agent = (isset($_POST['useragent'])) ? $_POST['useragent'] :
$_SERVER['HTTP_USER_AGENT'];
$wurflConfig = new WURFL_Configuration_XmlConfig($wurflConfigFile);
```

编辑这行代码。

device.php中，现在我们查看对应用户代理是否有一个输入的表单值（$_POST）中）。否则，默认为当前浏览器的用户代理。

device.php

试一试

1 保存所做的全部修改，并在一个Web浏览器中加载index.php文件。

第一次加载这个页面时，应该会看到你自己的浏览器的WURFL能力。

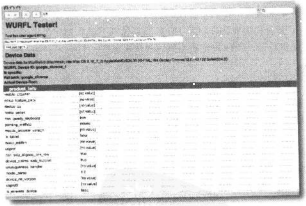
2 尝试几个移动浏览器用户代理。

在表单中输入一些移动浏览器用户代理，查看得到的能力。

```
Mozilla/5.0 (Linux; U; Android 2.2.1; en-us; DROIDX Build/
VZW) AppleWebKit/533.1 (KHTML, like Gecko) Version/4.0 Mobile
Safari/533.1 480X854 motorola DROIDX
```

```
Mozilla/5.0  (webOS/1.4.5;  U;  en-US)
AppleWebKit/532.2  (KHTML,  like  Gecko)
Version/1.0 Safari/532.2 Pre/1.0
```

试试这些用户代理。



现在可以输入任何user-agent串，并查看匹配设备的能力。

查找浏览器当前UA串的一个简便方法：http://whatsmyuseragent.com

Relax

是不是厌倦了输入user-agent串？

我们知道你可能已经受够了。可以在chapter5文件夹中找到一个ueful_user_agents.txt文件。这一章提到的所有用户代理都在这个文件中。你可以方便地从中复制粘贴。

把这些能力信息派上用场

我已经得到了大量能力信息。怎么利用它们做些有用的事情呢？



OK，来做点特别的。

现在要明确如何使用WURFL能力来确定是否应该在AcedIt!网站上显示那个紧急按钮。

噢，对啊，这才是我们原本想要做的。

使用WURFL帮助区分内容

只有当WURFL指出这是移动浏览器时（而不只是使用CSS媒体查询区分出窄屏幕），我们才会在浏览器上显示大大的红色紧急按钮。下面会利用WURFL能力信息来实现（不过，由于你在使用桌面浏览器，所以肯定看不到这种情况发生）！

成功之路

- [ ] 创建一个配置文件，并增加一些代码初始化一些WURFL对象。也就是说，得到当前设备的有关信息，以备使用。

上一个例子中我们已经了解了如何做到这一点。

- [ ] 询问WURFL我们关心的一个或多个特定能力。

- [ ] 在I'm Freaking Out！模拟页面中增加一些代码，根据能力值修改发布到不同设备的内容。

向WURFL问几个合适的问题

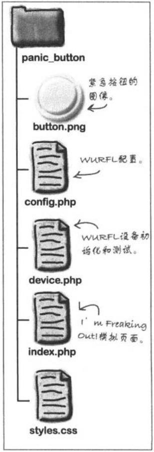
我们在165页上简单地认识了$device对象的getCapability方法。

这是一个很方便的方法。即使你对类和实例之类的东西一窍不通，得到这些能力信息也不太难，而且我们会告诉你怎么做。

要得到一个特定能力的值，如下所示：

动手做！

将config.php文件从explore目录复制到panic_button目录。



目录和文件结构

$value可能是设备能力值（一个字符串），如果没有值，则为NULL。

例如，'is_tablet'或'brand_name'

我们要问的能力名（一个字符串）。

```
$value = $device->getCapability($capability_name);
```

已经初始化并填充的WURFL CustomDevice对象。

稍后我们会在device.php文件中完成初始化和填充。

举几个例子：

这是一个移动设备吗？

```
$value = $device->getCapability('is_wireless_device');
```

```
$value = $device->getCapability('cookie_support');
```

这个浏览器（或设备）支持cookie吗？

需要说明，获取能力值时，不用担心这个能力在哪个组中。组是一个很好的组织概念，不过每个能力都有唯一的名字，可以直接获取。不需要使用能力组与WURFL交互。

想查看所有WURFL能力的信息？请访问http://wurfl.sourceforge.net/help_doc.php。

初始化设备，准备好信息

你会看到，我们在panic_button文件夹中为你提供了一些帮助。device.php文件已经包含了你需要的很多WURFL样板代码来完成后续工作。

与建立资源管理页面时不同，我们会使用另一种方法实例化设备。这里不再使用一个user-agent串并调用getDeviceForUserAgent方法，我们会告诉WURFL使用当前服务器请求中的值（包含在总是可用的PHP $_SERVER变量中）。

可以这样翻译，请为我们提供当前访问服务器的设备的相关数据。

```
$wurflManager = $wurflManagerFactory->create();
$device = $wurflManager->getDeviceForHttpRequest($_SERVER);
```

device.php

除了user-agent首部外，这个方法还会从另外几个HTTP首部获得提示，不过user-agent仍然是主要线索来源。

它是移动的吗？

现在我们已经有了WURFL设备。下面向它提问。

WURFL有一个很好的基本能力，名为is_wireless_device，它能回答这个基础问题，这是一个移动设备吗？所以看来我们应该这样做：

听上去就像一个是非(Boolean)问题。

```
$device = $wurflManager->getDeviceForHttpRequest($_SERVER);
$is_phone = $device->getCapability('is_wireless_device');
```

device.php

哈，对吗？

危险，威尔·鲁滨逊！

在PHP API中，所有能力值都作为字符串返回。这说明，看上去是Boolean型的能力is_wireless_device有3个可能的值： 'true'、'false' 或NULL（NULL表示WURFL根本没有对应这个能力的值）。

我们要更明确地指出到底是不是移动设备，否则值'false'会计算为true，非移动设备看上去就会像是移动设备。所以必须做一点修改，将之前写的最后一行代码替换为下面的新版本：

在PHP中，字符串 'false' 是个真值。如果你不清楚这是什么意思，没关系。只需使用下面的代码（而不是172页上最后的代码段）。

```
~~$is_phone = $device->getCapability('is_wireless_device');~~
$is_phone = ($device->getCapability('is_wireless_device') === 'true') ? true
: false;
```

这里的三等号操作符表示值必须完全等于字符串 'true'（而不是其他任意的真值），测试才能通过。

device.php

现在使用这个值

index.php文件包含了device.php文件，所以我们可以访问之前写的代码。我们希望在index.php中增加以下PHP条件。

```
<div id="content">
  <h1>I'm Freaking Out!</h1>
  <?php if ($is_phone): ?>
  <div id="panic_button">
    <img src="button.png" alt="HELP!" />
  </div>
  <?php else: ?>
  <h2>Help is only a phone call away.</h2>
  <div id="big_number">
  503-555-2939
  </div>
  <?php endif; ?>
  <p>Pre-test late-night jitters? A math problem that just
won't budge? Our expert on-call tutors are standing by to help
you through tough moments.</p>
</div>
```

index.php

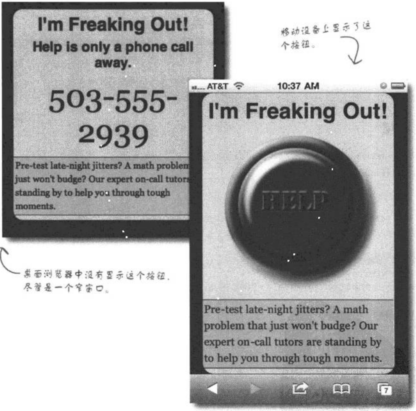
对于被认为来自移动设备的请求（根据is_wireless_device_capability判断），显示紧急按钮……

……否则，只显示电话号码，当然字体很大（通过一个CSS样式设置）。

试一试

完成171页和173页上对config.php和index.php的修改，保存这两个文件，然后在一个桌面浏览器和一个移动浏览器中分别查看得到的I'm Freaking Out! 页面。

移动设备上显示了这个按钮。



桌面浏览器中没有显示这个按钮，尽管是一个窄窗口。



WURFL闪亮登场

本周话题:

WURFL有什么用?

记者：嗯，WURFL，你到底能带来什么？

WURFL：很显然啊。你可以立刻区分出浏览器是不是移动的！

记者：要知道，还有很多其他方法也可以做到这一点。比如说，客户端检测，基本服务器端检测等。

WURFL：确定移动还是非移动，这只是小儿科。你还没有充分发挥我的潜力呢。

记者：好吧，我试试。

WURFL：那你按我说的做，在iPod Touch上看看这个页面。

记者：好的……（记者开始在移动设备抽屉里翻找，停顿很长时间之后）……哎呀，出现了大红按钮。

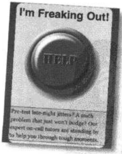
WURFL：这是不是说明了点什么……嗯，可能不是你想要的吧？你什么时候能用iPod打电话？

记者：没错。这个按钮出现在iPod Touch上一点意义都没有。现在我们又回到原点了。

WURFL：不，完全不是这样。就像我说的，你还没有完全发挥我的潜力呢。再深入挖掘挖掘我的能力。我想你会有一到两个新发现。

记者：不过，你有500多个能力啊！我怎么找到合适的那个呢？

WURFL：你是在抱怨我太过丰富了吗？唉，真是无语了，我简直没有对的时候！好吧好吧，给你个提示：按钮出现在移动设备上并不表示它真的会拨电话。你还没有连接呢，所以它只是个漂亮的图片，根本没有功能。你可能想到这一点了，再来看看我的能力。



iPod Touch上也显示了这个按钮……这没有任何意义。

Exercise

查看我们的资源管理页面，研究iPod Touch的能力（我不希望按钮出现在iPod Touch上），与Android Nexus S（我们希望在它上面显示按钮）的能力做个比较。你还可以看看桌面浏览器，做做比较。

你能找出一个能力帮我们区分移动（或小）设备与真正的手机吗？

iPod Touch UA

```
Mozilla/5.0 (iPod; U; CPU iPhone OS 4_3_2
like Mac OS X; en-us) AppleWebKit/533.17.9
(KHTML, like Gecko) Version/5.0.2 Mobile/8H7
Safari/6533.18.5
```

Nexus S UA

```
Mozilla/5.0 (Linux; U; Android 2.3.3; en-us;
Nexus S Build/GRI40) AppleWebKit/533.1 (KHTML,
like Gecko) Version/4.0 Mobile Safari/533.1
```

Exercise Solution

WURFL bearer（承载）能力组有一个漂亮的能力，名为has_cellular_radio。我们会查看这个能力的值来得出当前请求是不是来自一个手机（而不只是任意的移动设备）。

我们还会认识一个xhtml_make_phone_call_string能力，它会向我们展示如何为这个设备创建最好的可点击的电话号码链接。

利用WURFL让页面更聪明一些

通过测试has_cellular_radio的值，我们可以清楚地了解当前用户是不是在使用一个手机。我们还需要知道如何链接电话号码，使用户一旦单击这个链接，就能立即拨出电话。

xhtml_make_phone_call_string中的值可以为链接电话号码时应使用的语法给出一个提示，得到这个提示后，用户单击链接时，手机就知道要拨出电话。对于Nexus S，这个值是tel:。

如果has_cellular_radio能力值是"true"，而且xhtml_make_phone_call能力有一个有用的值，就可以认为这是一个手机！

对于iPod Touch，这个值是none。iPod Touch不能打电话。桌面浏览器也会返回none值。

通过结合xhtml_make_phone_call_string中的值和我们要拨的电话号码，就可以创建一个可用的链接。

用链接拨电话

移动手机浏览器会把链接中的某些URI模式识别为电话号码。用户单击这种链接时，会弹出一个提示框，要求确认他确实想拨出电话。

tel:是最常见的电话号码URI模式，适用于几乎所有现代智能手机。不过，使用xhtml_make_phone_call_string字段中的WURFL值没有坏处（反正你已经得到了这个值）。最后我们会得到：

tel:链接不需要国家代码(+1)。不过为了保证正确，最好加上国家代码。另外，这还会让你看起来很有“国际范”。

```
<a href="tel:+15035552939">Make a phone call</a>
```

紧急按钮：只针对手机

可以在device.php文件中另外包含一些测试，查看has_cellular_radio和xhtml_make_phone_call_string值。

首先，得到这些值。

我们不再需要这一行代码。

```
$is_phone       = ($device->getCapability('is_wireless_device') === 'true') ?
true : false;
$has_radio      = $device->getCapability('has_cellular_radio');
$phone_string = $device->getCapability('xhtml_make_phone_call_string');
```

device.php

现在来评判它们的值。

```
$phone_string = $device->getCapability('xhtml_make_phone_call_string');
$is_phone       = false;
if ($has_radio === 'true' && $phone_string && $phone_string !== 'none') {
  $is_phone = true;
}
```

device.php

Watch it!

'none'是一个有含义的值！

还记得173页曾经指出，PHP API中，CustomDevice>getCapability()可能返回一个字符串（如果提供了这个能力的值），也可能返回NULL（只有当WURFL根本没有提供这个能力值时才会返回NULL）。二者的区别非常重要。

对于WURFL能识别的类似iPod Touch和桌面浏览器等设备，xhtml_make_phone_call_string的值都是字符串 'none '。如果把它处理为PHP中的一个Boolean值，就会被评判为TRUE。

正是因为这个原因，我们不仅要测试$phone_string的存在，还要测试它的值不是 'none '。

there are no Dumb Questions

问：难道移动手机浏览器不会自动识别电话号码并帮你接通吗？

答：大多数情况下确实会。不过有两点要注意。对于我们目前的情况，最重要的一点是：我们的链接是一个图像，而不是一个电话号码。这里没有文本。手机很可能无法识别。

另外，不同手机准确识别电话号码的能力并不确定。Opera Mini通常会把加4位编码的邮政编码当作电话号码。试图向一个邮政编码拨电话会有严重的后果，你应该能想见。

另外要记住，我们的电话号码格式相当规范（+15035552329）。但有人可能希望在链接文本中显示一个稍有不同的字符串，还自信地认为链接本身会正确地格式化。

问：为什么用户点击一个电话号码链接之后必须弹出那个确认框？为什么不直接拨号？

答：这是为了保护用户不上当受骗。可见的电话号码链接可能有一个号码，而链接本身却有另一个号码。一些可怜的家伙以为他们在打电话取消订阅当地报纸，但实际上可能在给某个骗子打电话。

在这里，我们甚至没有显示电话号码，有人会认为这并不是一个最佳实践。我们也可以使用一个CSS背景图像而不是一个在线图像，让电话号码作为文本显示在链接中，并在CSS中使用一个很大的负边距值隐藏这个文本。

问：用英语解释好吗？我大学没有主修CSS。

答：使用一个很大的负左边距值（例如margin-left: -10000px）是一种让文本从左边消失不可见的方法。我们希望这个文本出现在距左边10000像素的某个位置，就像漂浮在办公室的空气中。当然，这样确实有点怪。

问：这么说，我们认为如果浏览器的xhtml_make_phone_call_string有一个非none值，就说明它在一个可以拨电话的移动手机上，不过怎么知道呢？

答：WURFL有一些“智能”数据，但还不至于那么聪明（没有人能那么聪明）。没有一种明确的方法可以绝对确信一个用户在一个有效的蜂窝网络上（而不是WiFi或者其他网络），否则没有其他方法能够防止拨出电话。不过，实际上这对我们很有用。

问：我们不能只测试xhtml_make_phone_call_string值不为'none'吗？测试has_cellular_radio是不是多余？

答：你可能有点想法，不过也许只是自作聪明。

问：你说自作聪明，这只是在搪塞我，不过我确实想更多地了解WURFL对象和API……

答：嘿，我们真心建议你去看看API中的PHP类，了解它们是如何一起工作的。如果你安装了API，应该已经有这些文件了，快去看看吧！



iPod Touch上再没有出现按钮了！不过iPhone上按钮仍然正常显示！



Jim：这会失控的！在一个更复杂的项目中，或者甚至在AcedIt!网站中，如果增加更多移动细节，看起来要在代码中到处使用分散的单个WURFL能力，这会导致大量问题。我的意思是说，如果最后在要一个模板中测试一个能力，而在另外一个地方测试另一个能力……如此等等，那可太糟糕了！

Frank：我同意。看来我们要为每一种可能的能力组合发布一个不同的网站（可能只是稍有不同），这对于考虑和测试来说绝对是个恶梦。

Jim：没错，这样只可能得到一种“通心粉式”的代码。

Frank：不过，我不希望摒弃整个概念。尽管我愿意尝试在客户端完成特性检测，而且希望我们的内容尽可能提供响应性，但看来利用WURFL之类的设备数据库，我们可以更深入地了解一些细节，而这些通过其他途径可能无法得到。

Jim：不过，如何把所有这一切都赶到“畜栏”里，而不是分散得到处都有？以免我们发狂！

Frank：看起来，如果能对设备逻辑分组，而不是逐个地测试能力，我们就能保持淡定了。还记得吗？最近我们曾经花了一些时间考虑如何判断要支持哪些设备，我们的做法是选择一些特性，并对不满足要求的手机划定界线。

Jim：没错。我还对那些练习念念不忘呢！

Frank：看来我们可以更进一步，可以创建一堆设备，不过要归类到我们决定支持的设备组中。

Jim: 有意思……我想你可能有办法了。

Exercise

用177页上的内容修改173页的代码，修改了device.php之后，你可能会发现你没有一个iPod Touch来测试这些修改。

要查看iPod Touch会看到什么，可以编辑device.php文件。使用getDeviceForUserAgent()方法而不是getDeviceForHTTPRequest()方法，它会给出一个iPod Touch用户代理。

在第175页可以找到一个iPod Touch用户代理。

答案见第180页。

放牧设备

设备类就像创建了一种逻辑畜栏，可以把有某些共同点的设备赶到这个畜栏中。

牧民可能把花点奶牛关在一个牛圈里，把大块头的马放在另一个马厩里，而小猪仔则放在另外一个猪圈里（就好像我们当过牧民一样），类似的，我们也可以把设备分类到虚拟的设备类小房间里。

设备类是定义一组设备（及其浏览器）的共同特性的一个抽象集合。

一旦分类，继续前进

一旦牧民发现某个牲畜属于这3组之一，他就能做出进一步的决定，而不用查看更小的细节。他可以给马喂燕麦，而不用停下来反复确认它们的腿是否有一定长度或者有长长的马尾，或者是不是能支持PNG图像（当然，所有人都知道马并不支持透明PNG）。

他已经知道它们是马，因为它们都在马厩里。他不用单独喂养每一个牲畜（这需要跟踪大量马料）。不过，与此同时，他也不会给小猪喂燕麦。





初始化WURFL设备时，通过使用设备的用户代理可以测试向一个给定的设备或浏览器发布哪些内容。

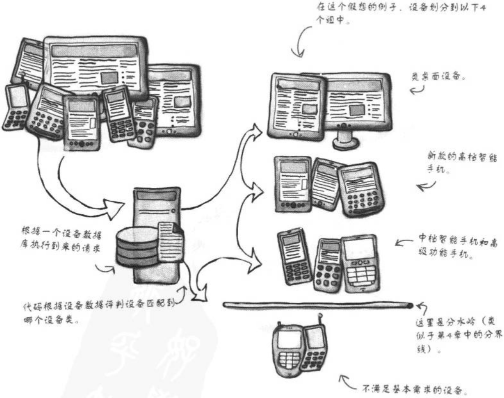
```
$wurflManager = $wurflManagerFactory->create();
$device = $wurflManager->getDeviceForHttpRequest($_SERVER);
$user_agent = "Mozilla/5.0 (iPod; U; CPU iPhone OS 4_3_2 like Mac
OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2
Mobile/8H7 Safari/6533.18.5";
$device = $wurflManager->getDeviceForUserAgent($user_agent);
```

device.php

设备类

之前我们已经了解到，可以根据一个设备数据库执行到来的请求，来得到设备信息。通过将对影响当前网站的能力和值进行分组，可以让内容适应整个设备类，而不是追踪每一个能力的值。

这些设备类定义（将转换为代码）将设备划分到某个组。一旦对设备分组，我们就能采取相应的行动，而不用跟踪每一个单个的能力。



业务拓展



嘿，伙计们。那个紧急按钮的问题处理得不错！能不能再帮我们一次，我们正在构建一个大型的新网站，能帮我们对它完成移动优化吗？

扩大AcedIt!的业务范围

AcedIt!开始向学生们销售学习辅导资料，并为此建立了一个独立的网站。可以看到，类似闪卡、练习测验小册子、参考书之类的东西很好卖，不过目前要购买这些东西，唯一的办法就是通过印刷出版的图书目录。

该有所改变了。AcedIt!建立了一个专门针对学习辅导的新网站，不仅允许用户在线购买所有实物产品，还引入了一些在线产品，如闪卡，可以直接在网站上交互式使用。

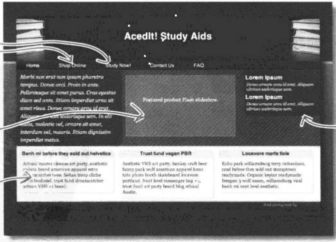
AcedIt!的开发人员在用JavaScript和其他标准Web技术构建这些特性。

提前看看到底要提供一个怎样的网站

现在还太早。Logo设计还没有最后敲定，内容还在审定之中，另外设计也还只是处于简单的模拟阶段。不过先来看看AcedIt!到底要建立怎样的网站：

AcedIt!的Web开发人员已经把这个新网站主页的早期模拟版本发给我们。

用户现在可以在线购买辅导材料，还可以在线学习，使用这个网站上的交互式产品。



AcedIt!将通过一个漂亮的Flash视频宣传产品。

这些广告对应相关服务，如文章校对和当地研究小组。

主页上会摘录公司博客的文字。

戴上移动眼镜评判这个主页

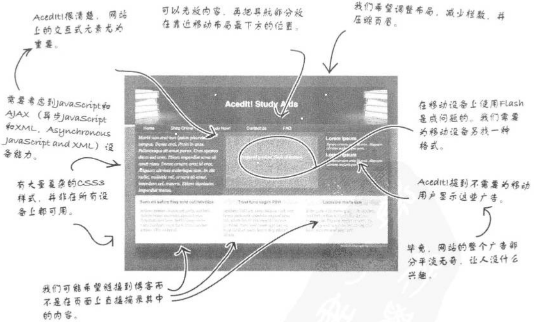
这个新的学习辅导网站涵盖的范围要比之前那个紧急按钮页面大得多。如果逐个能力地规划开发工作，会让我们发疯的。要制定开发方法，使这个网站适用于多种不同的用户，一种办法就是使用设备类对不同的体验分组。

好吧，知道了。我们现在只有一个粗略的模拟页面作为开发起点。不过这个项目涵盖的范围……最终肯定会更广。

结合设备数据和逻辑分组

定义设备类的过程与我们在第4章做的工作相关。要查看当前任务，明确哪些方面比较重要，念几句咒语，找出与一般原则相左的情况。

我们知道，“使用设备类对体验分组”这句话意思很含糊。下面首先再来看看这个模拟页面，根据我们对移动Web特性和约束的了解明确AcedIt!的优先级。

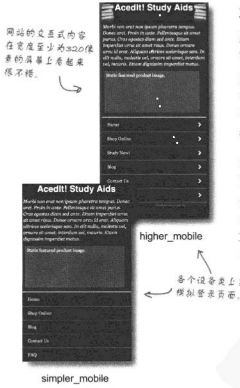


将需求分组为多种移动体验

完全的交互式网站体验有一些很高的要求。移动设备需要提供JavaScript、HTML和CSS支持，智能手机上较新的时髦浏览器通常都会提供这些支持。具体来讲，AcedIt!开发人员所针对的就是这样一类移动设备，要求运行基于WebKit的浏览器，并为屏幕宽度不小于320像素的设备而设计。

不过这并不表示其他人会被冷落。网站也欢迎那些比较小的、不那么时兴的手机。这些手机的用户可以在在线商店购买实物产品，还可以访问网站的其他板块。只是他们无法得到一些高级的特性，如完美的交互式体验或核心CSS。

我们并不只是向满足最低要求的设备发布内容，也不会干脆排除那些不满足要求的设备，下面创建两个不同的网站，分别适应各组设备。



更丰富的移动体验

通过创建一个设备类来涵盖更丰富的移动体验需求，可以设计一些相当漂亮的东西，而不用担心低档设备缺乏某些支持。

通过指定这一类设备必须有一个基于WebKit的浏览器，可以相当自信地使用梯度之类的CSS特性。我们还可以利用某种程度的高级JavaScript支持。

面向简单手机的简化体验

如果设备屏幕比较窄，能力不太强，就要更精简一些了。不能假设它支持所有现代浏览器的特性。

这个更简单的布局不再提供在线商品的链接，不过用户可以在在线商店里购物。这个版本在大约176像素宽的屏幕上可以工作得很好，如果屏幕小于这个宽度，就很难看清商店里商品的图像了。

现在我们对两个移动设备类higher_mobile和simpler_mobile已经有了大致的认识。

设备类排排坐

现在我们对要构造的两个移动设备类已经有了大致的认识，还有什么呢？

噢，对了，还有平板电脑

AcedIt!的开发人员还为他们的在线商品提供一个针对平板电脑优化的超级漂亮的触摸式界面。这个工作还没有完成，不过他们希望现在能识别平板电脑，为网站的最后发布做好准备。

在哪里划分界线

对于AcedIt!开发人员来说，他们的访问者应该可以在线购物，或者可以使用在线商品，这一点很重要。如果由于设备限制，导致访问者既不能在线购物也不能使用在线商品，就认为他们使用的设备和浏览器是不支持的（unsupported）。

公司的电子商务软件要求提供cookie支持，还要有一点JavaScript支持（远不如在线交互式商品所需的JavaScript支持）。另外，为安全起见，还强制要求设备支持SSL。

绝对必要

根据优先级和需求，看起来以下几点是最基本的要求。移动设备必须“支持”以下特性：

- SSL支持。
- Cookie支持。
- 至少有一些JavaScript支持。
- 分辨率至少为176px。

我们的设备类阵容





设备类闪亮登场

本周访谈：

抽象到具体……到底什么是设备类？

有点困惑的Web开发人员（以下简称为SCWD）： 我实在搞不清这个设备类到底是什么。一堆能力……与某种体验关联？

设备类： 我承认，掌握我是有点难度。如果我是一幅画，我就是那种抽象画。我是一个概念，一种组织共同事物的思路，这样我们可以只创建几类网站，而不是创建上百万个不同的网站。

SCWD： 这么说，设备类就是一组WURFL能力喽？

设备类： 别那么快下结论。要记住，我是一个抽象概念。尽管这里我们使用了WURFL，不过实际上并不要求你使用WURFL。其实甚至没有必要非得使用设备数据库。

SCWD： 开始有点不明白了。你能帮我解释一下吗？

设备类： 好吧。下面我们一起来做个计划。对于AcedIt!，共有5个设备类。

SCWD： 这是不是意味着我们要为每一个设备类建立一个网站版本？听上去可是个大工程啊。

设备类： 不，我们只强调它们的不同。例如，启用新的针对平板电脑优化的触摸式闪卡界面之前，网站的平板电脑设备类版本与桌面版本只有一点不同：它不会使用Flash内容。

SCWD： OK，这么说我们有一个桌面设备类，它的含义很明确……还有平板电脑设备类，只是稍有差别……嗯，我看看。那么higher_mobile和simpler_mobile有什么区别呢？

设备类： 我们都知道，higher_mobile表示这一类设备有较大的屏幕，另外有能力很强的基于WebKit的浏览器，我们可以很自信地认为它们有足够的能力处理网站中强度更大的交互式特性。

simpler_mobile正相反，这一类设备只有比较窄的屏幕，另外浏览器可能不太先进。我们只能给它们比较小的图像（相应的，带宽也比较小）。我们知道它们只支持一点JavaScript，不过可以提供一些不太华丽的内容。

SCWD： 等一下，我们怎么知道simpler_mobile设备支持JavaScript呢？

设备类： 我们会测试至少有一点点JavaScript支持。如果设备没有在页面加载之后修改DOM的能力，就会完全撇入unsupported设备类。

SCWD： 那么接下来要做什么呢？

设备类： 我们需要把适当的WURFL能力和值映射到我们想要创建的设备类。有了设备类的逻辑表示之后，再创建代码来完成一些具体的测试和归类。

池塘谜题



来为我们的设备类找出正确的WURFL能力和值。你的任务是从池塘里取出WURFL能力和值，把它们放入空的设备类中。同一项不能使用多次。

higher_mobile

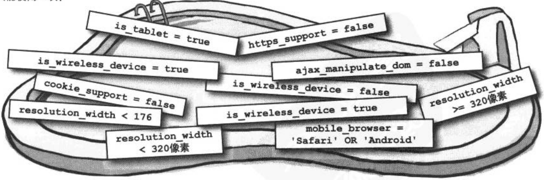
desktop

tablet

lesser_mobile

unsupported

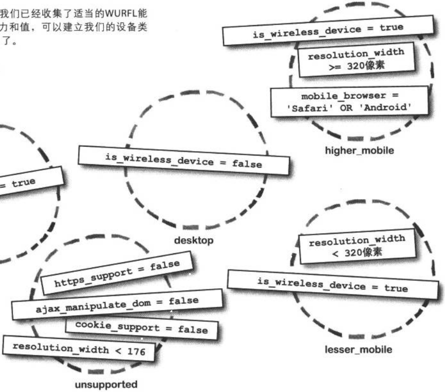
提示：池塘里的每一项只能使用一次！

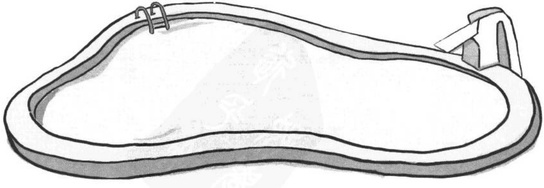


池塘谜题答案



现在我们已经收集了适当的WURFL能力和值，可以建立我们的设备类了。





there are no Dumb Questions

问：为什么要强调基于WebKit的浏览器？开发人员不喜欢那些漂亮的移动浏览器吗？

答：在很多移动Web开发人员看来，基于WebKit的浏览器相当高级，也很一致。有WebKit渲染引擎的浏览器一般都能很好地支持HTML5、JavaScript和CSS3。

注意上面这句话中的“一般”这个词。问题在于，对于基于WebKit的浏览器有多一致还存在误解。一个让人遗憾的事实是：这方面还相当混乱。

移动平台战略家Peter-Paul Koch在他的博客QuirksBlog (http://bit.ly/uWnFLa)中发表了一篇分析相当全面的博文“There is no WebKit on Mobile”，清楚地指出了假设移动WebKit浏览器一致性存在的种种问题。

问：这么说，移动WebKit实际上并不怎么样，为什么我们要让设备类基于它？

答：不论是否一致和可靠，基于WebKit的浏览器是开发人员完成开发和进行测试的平台。项目已经进行到这个阶段，再抛开它就为时太晚了。对于这个项目，我们还是沿着这个方向继续前进吧。

问：为什么测试移动浏览器名为字符串 'Safari' 或'Android' '就等于查找所有WebKit浏览器？

答：应该还记得第3章曾经说过，user-agent串很聪明。关于WebKit，Apple公司历史以来一直都在使用沿袭下来的user-agent串。写这本书时，所有已知的WebKit浏览器移动版本的浏览器名都是“Safari”或“Android”（是的，甚至黑莓、诺基亚手机和一些不知名的手机上的WebKit浏览器也是如此）。

问：如果我在项目中使用设备类，都必须建立5个设备类吗？

答：当然不是。你可以有10、4个，甚至一个都没有。设备类越少意味着复杂性越低。设备类越多说明差别越多。

问：higher_mobile和simpler_mobile的名字有什么说法吗？有没有我需要了解的命名约定？

答：嗯，我们只是随口这样取名的。这样听上去起码含义是正确的。这里使用下划线的目的只是为了能更容易地把名字转换为代码。你可以对设备类取你喜欢的任何名字，只要合理就行。

问：既然有了设备类，我再也没办法测试单个的设备能力了吗？

答：当然还可以测试单个的能力。实际上，对于前面的紧急按钮问题，就采用了这种解决方案：我们测试了一个非常特定的能力的值。对于同一个设备类中的不同设备，这个值就可能不同（例如，iPhone与iPod Touch相比，这个值就不同）。

问：一个设备不能匹配多个设备类匹配吧？

答：不能。我们需要很仔细地安排设备类测试代码的顺序。第一个匹配的设备类就是我们指定的设备类。

问：我在代码示例中总看到CustomDevice对象，这到底是什么？

答：CustomDevice是WURFL API中表示设备及其特性的一个类的类名。

问：看来设备类不只是一个移动Web概念。

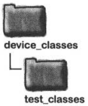
答：当然不是！内容和布局调整是整个Web都要考虑的问题。这可不是移动Web专属的概念。

问：OK，太好了。我已经把各个设备类的能力和所需的值放到一些圆圈里了。现在做什么？

答：翻开下一页，你可真性急。下面就要开始把这些设备类转换成代码。

实际演练

现在要把我们分组的抽象概念转换为真正的代码，能够对设备进行检测，归入适当的设备类。下面开始归类吧！



不过，首先……

找到chapter5中的device_classes文件夹，然后找到它下面的test_classes文件夹。找到了！这就是我们现在的演练场。

你可能感觉这个过程很熟悉：我们需要建立一个配置文件，从而能初始化WURFL。先从panic_button文件夹把config.php文件复制到device_classes/test_classes。



开始吧！

我们要完成以下工作：

- ☐ 编写一个(PHP)函数，测试各个设备类定义中我们关心的单个设备能力。
- ☐ 将各个设备类的能力需求转换为代码中的条件(if/then)语句，从而能确定当前设备应该分配到哪个设备类。
- ☐ 编写一个简单的测试页面（类似于我们前面编写的WURFL资源管理页面），查看不同的用户代理会归类到哪个设备类。

熟悉匹配函数

你会发现已经为你写好了这个匹配函数，就放在device_classes.php中。创建各个设备类的测试时可以直接使用这个函数。

使用这个匹配函数之前，有几点需要告诉你。先退一步，请看下面给出的一些PHP代码。

成品 PHP

要测试的能力名。

比较操作符。

测试所要比较的值。

确保设备对象存在。

从设备得到能力值。

根据我们关心的值测试比较设备的当前值。

```
function device_match($capability, $comparison, $value) {
  global $device;
  if (!$device) {
    return FALSE;
  }
  $device_value = $device->getCapability($capability);
  switch ($comparison) {
    case '==':
    case '===':
      return ($device_value === $value);
    case '!=':
    case '!==':
      return ($device_value !== $value);
    case '>=':
      return ($device_value >= $value);
    case '<=':
      return ($device_value <= $value);
    case '>':
      return ($device_value > $value);
    case '<':
      return ($device_value < $value);
    case 'LIKE':
      return (strpos($device_value, $value) !== FALSE);
    case 'NOT LIKE':
      return (strpos($device_value, $value) === FALSE);
    default:
      return FALSE;
  }
}
```

这个函数返回一个TRUE或FALSE(Boolean)结果，表示给定测试是否通过。

device_classes.php

这个switch语句在做什么？

$device_value是当前设备中我们关心的那个能力的值。

```
switch ($comparison) {
    case '==':
    case '===':
      return ($device_value === $value);
    case '!=':
    case '!==':
      return ($device_value !== $value);
    case '>=':
      return ($device_value >= $value);
    case '<=':
      return ($device_value <= $value);
    case '>':
      return ($device_value > $value);
    case '<':
      return ($device_value < $value);
    case 'LIKE':
      return (strpos($device_value, $value) !== FALSE);
    case 'NOT LIKE':
      return (strpos($device_value, $value) === FALSE);
    default:
      return FALSE;
}
```

$value是我们想要比较的值。

取决于提供的比较操作符，采用不同方式完成这些值的比较。

利用LIKE和NOT LIKE允许可以完成块（子串）比较。

如果$comparison不是一个可识别的比较操作符，测试失败。

举几个例子

如果设备的resolution_width值为240像素或更小，这个测试通过（返回TRUE值）。

```
device_match('resolution_width', '<=', '240');
```

如果这个设备的is_wireless_device值是'false'，这个测试通过。

```
device_match('is_wireless_device', '===', 'false');
```

如果设备的mobile_browser值包含字符串'safari'，这个测试通过。

```
device_match('mobile_browser', 'LIKE', 'Safari');
```

使用匹配函数测试能力

现在我们有了一个可以用来测试能力的函数。下面把前面要求的必要能力转换为测试。

- ☑ 编写一个(PHP)函数，测试各个设备类定义中我们关心的单个设备能力。
- ☐ 将各个设备类的能力需求转换为代码中的条件(if/then)语句，从而能确定当前设备应该分配到哪个设备类。
- ☐ 编写一个简单的测试页面（类似于我们前面编写的WURFL资源管理页面），查看不同的用户代理会归类到哪个设备类。

提醒：我们最后建立的设备类分别是desktop、tablet、higher_mobile、simpler_mobile和unsupported。各个设备类要测试的能力已经在188页的池塘谜题中确定。



我们已经为你编写了匹配函数，不过现在该你做些工作了。下面哪些测试属于哪些设备类?

```
.................... if (device_match('resolution_width', '>=', '320'))

.................... if (device_match('resolution_width', '<', '176'))

.................... if (device_match('is_wireless_device', '===', 'false'))

.................... if (device_match('mobile_browser', 'LIKE', 'Safari') ||
                         device_match('mobile_browser', 'LIKE', 'Android'))

.................... if (device_match('cookie_support', '===', 'false'))

.................... if (device_match('is_tablet', '===', 'true'))

.................... if (device_match('ajax_manipulate_dom', '===', 'false'))

.................... if (device_match('https_support', '===', 'false'))

.................... if (device_match('resolution_width', '<', '320'))
```

Exercise Solution

我们已经完成了这些测试的归类！现在该把它们转换成真正的PHP代码了。继续前进吧！

感觉这可能是对应桌面设备类的一个测试，不过通过查看is_wireless_device能力就能立即排除桌面设备类……

higher_mobile	`if (device_match('resolution_width', '>=', '320'))`
unsupported	`if (device_match('resolution_width', '<', '176'))`
desktop	`if (device_match('is_wireless_device', '===', 'false'))`
higher_mobile	`if (device_match('mobile_browser', 'LIKE', 'Safari') || device_match('mobile_browser', 'LIKE', 'Android'))`
unsupported	`if (device_match('cookie_support', '===', 'false'))`
tablet	`if (device_match('is_tablet', '===', 'true'))`
unsupported	`if (device_match('ajax_manipulate_dom', '===', 'false'))`
unsupported	`if (device_match('https_support', '===', 'false'))`
simpler_mobile	`if (device_match('resolution_width', '<', '320'))`

该汇总代码了……

Sharpen your pencil

熟悉PHP代码吗？看看你能不能填入这里缺少的代码块。这些是已经分组的设备类测试，已经万事俱备了。

```
$device_class = NULL;

if (device_match('is_wireless_device', '===', '..........')) {
  $device_class = 'desktop';
}
else if (device_match('https_support', '........', '..........') ||
          device_match('.......... _support', '........', '..........') ||
          device_match('ajax_manipulate_dom', '===', 'false') ||
          device_match('..............................', '........', '176')) {
  $device_class = '..................................';
}
else if (device_match('is_tablet', '===', '..........')) {
  $device_class = '.................';
}
else if (device_match('is_wireless_device', '===', 'true') &&
         device_match('resolution_width', '>=', '........') &&
          (device_match('mobile_browser', '........', 'Safari') ||
           device_match('.........................', 'LIKE', 'Android') ))
{
  $device_class = '...............................................';
}
else if (device_match('is_wireless_device', '===', '........') &&
        device_match('................................', '<', '..........')) {
  $device_class = 'simpler_mobile';
}
```

Sharpen your pencil Solution

好了，我们的测试已经完成！

```
$device_class = NULL;

if (device_match('is_wireless_device', '===', 'false ')) {
  $device_class = 'desktop';
}
else if (device_match('https_support', '===', ' false ') ||
          device_match('cookie_support', '===', 'false ') ||
          device_match('ajax_manipulate_dom', '===', 'false') ||
          device_match(' resolution_width', '<', '176')) {
  $device_class = 'unsupported';
}
else if (device_match('is_tablet', '===', 'true ')) {
  $device_class = ' tablet ';
}
else if (device_match('is_wireless_device', '===', 'true') &&
         device_match('resolution_width', '>=', ' 320') &&
           (device_match('mobile_browser', ' LIKE', 'Safari') ||
            device_match('mobile_browser', 'LIKE', 'Android') ))
{
  $device_class = 'higher_mobile';
}
else if (device_match('is_wireless_device', '===', 'true ') &&
         device_match('resolution_width', '<', ' 320')) {
  $device_class = 'simpler_mobile';
}
```

首先也是最重要的：这是一个移动浏览器吗？

确保这个设备满足我们的基本需求……

可能是一个平板电脑吗？

或者它的浏览器和分辨率很大吗？

或者也许它还不错，不过不够前沿。

好了！3个任务我们已经完成了两个！就快大功告成了！

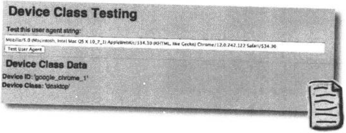
- [x] 编写一个(PHP)函数，测试各个设备类定义中我们关心的单个设备能力。
- [x] 将各个设备类的能力需求转换为代码中的条件(if/then)语句，从而能确定当前设备应该分配到哪个设备类。
- [ ] 编写一个简单的测试页面（类似于我们前面编写的WURFL资源管理页面），查看不同的用户代理会归类到哪个设备类。

最后冲刺

现在来完成这个设备类测试项目的最后一步。我们想建立一个页面，其中包含一个简单的表单，输入一个用户代理后，它就能给出这个用户代理匹配的设备类。由于这与之前在WURFL资源管理页面上所做的工作极其类似，所以我们已经帮你完成了这个页面（太棒了）。

我们已经编写了这个简单的测试表单，放在index.php中。

这个页面会显示设备ID以及对应所输入的用户代理得出的设备类。



index.php

试一试

1 在device_classes.php文件中加入代码。

把196页上的测试代码增加到这个文件中，放在device_match函数后面，并保存。

2 在一个浏览器中查看index.php文件。

默认地，页面会向你显示为当前浏览器分配的设备类。

3 测试另外一些不同的用户代理。

把这里的用户代理输入到表单域中，查看它们分别分配到哪个设备类。

```
Mozilla/5.0 (PlayBook; U;
RIM Tablet OS 1.0.0; en-
US) AppleWebKit/534.8+ (KHTML,
like Gecko) Version/0.0.1
Safari/534.8+
```

```
PantechP2020/JIUS05172010R;
Mozilla/5.0 (Profile/MIDP-2.0
Configuration/CLDC-1.1; Opera
Mini/att/4.2.19039; U; en-US)
Opera 9.50
```

```
BlackBerry8330/4.5.0.77 Profile/
MIDP-2.0 Configuration/CLDC-1.1
VendorID/105
```

```
BlackBerry9300/5.0.0.794
Profile/MIDP-2.1 Configuration/
CLDC-1.1 VendorID/245
```

嗯，来看看……到底怎么样？

不错！

Device ID: 'rim_playbook_ver1'
Device Class: 'tablet'

这个RIM Playbook正确地标识为一个平板电脑。

不错！

Device ID: 'pantech_p2020_ver1'
Device Class: 'simpler_mobile'

这个Pantech智能手机归为simpler_mobile，因为它的屏幕比较小。

不错！

Device ID: 'blackberry8330_ver1_sub45077_105'
Device Class: 'unsupported'

这个BlackBerry的浏览器太老了（4.5版本），所以不允许在页面加载之后管理DOM。

也就是说，它缺乏必要的某种程度的JavaScript支持。

哎呀！

Device Class Data

Device ID: 'blackberry9300_ver1'
Device Class: ''

哎呀！没有为这个用户代理指定任何设备类！

为什么BlackBerry 9300 (Curve)的用户代理没能匹配任何设备类呢？

看来出了点问题

问题是，这个Curve的分辨率是320像素（这说明它很宽，不能归入simpler_mobile类），但是又没有WebKit浏览器（说明它不能算作是higher_mobile）。其他有较高分辨率但没有WebKit浏览器的设备都存在这个问题。

我们需要修正这个问题，确保我们的设备类没有逻辑漏洞，以免设备落入。

填补缺口

要修正这种有高分辨率但没有WebKit浏览器的情况，我们有3种选择：

1. 可以创建一个新的设备类来处理这种组合。
2. 可以修改higher_mobile设备类，允许有其他非WebKit浏览器。 ← 按分辨率分组。
3. 可以编辑simpler_mobile设备类，并删除分辨率最大为240像素的限制。 ← 按浏览器分组。

上述各种选择都是完全合法的。我们必须确定哪一种方案对AcedIt!学习辅导网站最为理想。



Joe：真的需要再增加一个设备类吗？看上去太麻烦了吧。

Frank：我同意。我们需要在细节差异和设备类数量之间达到某种平衡。感觉现在的设备类数量应该正合适。

Joe：那么，现在该怎么办？

Frank：嗯。问题在于有高分辨率但没有WebKit浏览器的设备被绕过去了，它们没有分配到任何设备类。

Joe：这里实际上是不是有两个问题？一个问题是，正像你说的，我们没有考虑这些设备应该得到怎样的体验。另外还有一个问题，这就是我们没有留出一个可以作为退路的默认设备类。看起来设备检测很可能会出问题，或者可能还会出现我们没有想到的其他问题。我想我们需要某种类似于安全网的设备类。

Frank：说得好。对于第一个问题，如果不再增加一个新的设备类，就需要填补这个缺口。问题在于：这里什么更重要，是设备分辨率还是浏览器能力？

Joe：AcedIt!非常关注网站的交互性元素，这说明浏览器能力可能更重要。不过，与此同时，我们还打算向那些低分辨率设备发送较小的图像。

Frank：我想你对AcedIt!优先级的考虑是对的。如何折中呢？我们可以更新设备类，使得simpler_mobile类没有分辨率上限。

这说明一些较高分辨率的设备也会得到比较简单的界面，不过如果我没有记错，处理触摸式闪卡的开发小组在使用一个面向WebKi浏览器的框架。嘿，实际上，我想这正是刚开始在higher_mobile类中测试WebKit浏览器的原因。嗯，我得睡会儿了，脑子有点不够用了。

Joe：等等，还有一个问题，图像呢？

Frank：我觉得如果为所有移动设备提供移动优化的图像（不大于320像素），应该很不错。这样我们就能使用响应式图像技术解决设备最后的问题。

填补设备类测试的缺口

我们需要对设备类测试做以下修改：

删除simpler_mobile的这个分辨率限制。

```
  $device_class = 'higher_mobile';
}
else if (device_match('is_wireless_device', '===', 'true') &&
        device_match('resolution_width', '<', '320')) {
  $device_class = 'simpler_mobile';
}
else {
  $device_class = 'desktop';
}
```

注意：只删除这里的一个括号。

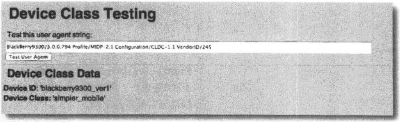
增加一个作为退路的默认设备类。

device_classes.php

试一试

修改device_classes.php文件，在测试表单中重试197页上的user-agent串。

哈！原先有问题的黑莓用户代理现在与simpler_mobile设备类匹配。

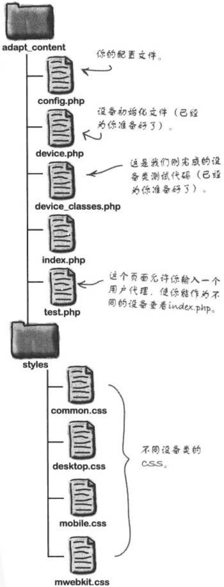


用设备类真正做些事情

现在该用我们建立的设备类做些事情了。

目标：根据设备类向不同的用户发布不同的登录模拟页面。

我们将使用前面的代码把设备划分为设备类，并向不同的设备发布当前AcedIt!学习辅导模拟网站的不同版本，这里要使用我们在184页设计的框架。



动手做！

将配置文件从上一个练习复制到adapt_content文件夹。唉呀，又出问题了！不过我们保证，这是最后一次。

这里处理的HTML和CSS相当基础，AcedIt!还处于这个新网站开发的的早期阶段，所以我们还在处理相当基本的模拟网站。

对unsupported设备的处理类似于桌面浏览器……至少是现在

向unsupported设备类发布的内容与桌面内容基本上相同，不过我们会去除网站中交互式闪卡部分和在线商店的链接。另外Flash内容也不会发布到unsupported设备。

Relax

有点太晚了，我们已经累了：为了节约你的时间，我们已经帮你做了很多工作。

后面几页会分析我们是如何做的，不过现在先告诉你在哪里可以找到这些已经完成的代码。

Sharpen your pencil

作为起点，index.php包含了各种版本的内容（对应所有设备类）。现在该明确会向各种不同设备类的用户发布（或不发布）哪一部分内容。对于以下的各个序号，在右边找到相应的标记，并填入适当的设备类。

1. 对 .... 设备类使用这个DOCTYPE 。

2. 对 .... , ... 和 .... 设备类使用这个样式表。

3. 对 .... 和 .... 设备类使用这个样式表。

4. 对 .... 设备类使用这个样式表。

5. 对 .... , ... 和 .... 设备类显示这个版本的导航。

6. 对 .... 设备类不要显示这两个链接。

下面是index.php文件的标记。用它回答左页上的问题。

index.php

```
  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN"
❶    "http://www.w3.org/TR/xhtml-basic/xhtml-basic11.dtd">
  <!DOCTYPE html>
  <html>
  <head>
    <title>AcedIt! Study Aids</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1,
  maximum-scale=1" />
    <link rel="stylesheet" type="text/css" href="../assets/common.css" />
❷   <link rel="stylesheet" type="text/css" href="../assets/desktop.css" />
❸   <link rel="stylesheet" type="text/css" href="../assets/mobile.css" />
❹   <link rel="stylesheet" type="text/css" href="../assets/mwebkit.css" />
  </head>
  <body>

  <div id="header">
    <h1>AcedIt! Study Aids</h1>
  </div>
  <div id="navigation">
❺   <ul>
      <li><a href="#">Home</a></li>
      <li><a href="#">Shop Online</a></li>
      <li><a href="#">Study Now!</a></li>   ❻
      <li><a href="#">Contact Us</a></li>
      <li><a href="#">FAQ</a></li>
    </ul>
  </div>
```

注意……Study Now部分是基于JavaScript的交互式闪卡特性。

下一页继续。

Sharpen your pencil

噢，你以为做完了？还没有呢，不过至少你已经完成一半了！

7 向 .... 设备类发布Flash内容。

8 对 ....，..，
.... 或 .... 设备类不要使用Flash。

9 对 .... 或 ....
设备类不要显示这些部分。

10 向 .... 和
.... 设备类显示这个版本的导航。

11 向 .... 设备类隐藏交互式闪卡的链接。

index.php

```
<div id="intro">
<p>Morbi non erat non ipsum pharetra tempus. Donec orci. Proin in ante.
Pellentesque sit amet purus. Cras egestas diam sed ante. Etiam imperdiet
urna sit amet risus...</p>
</div>
<div id="feature">
  <div id="featured_product">
  ❼ <p>Featured product Flash slideshow.</p>
  ❽ <p>Static featured product image.</p>
  </div>
</div>
<div id="ads">
  ...
</div>
❾<div id="fromtheblog">
  ...
</div>
<div id="navigation">
  <ul>
❿   <li><a href="#">Home</a></li>
    <li><a href="#">Shop Online</a></li>
    <li><a href="#">Study Now!</a></li>
    <li><a href="#">Blog</a></li>        ⓫
    <li><a href="#">Contact Us</a></li>
    <li><a href="#">FAQ</a></li>
  </ul>
</div>
<div id="footer">
  <p>Current device class: <?php print $device_class; ?></p>
</div>
</body>
</html>
```

这可能很明显，不过这些只是占位符，开发Flash视频和静态图像之后它们将被替换。

为了节省这一页的篇幅，我们删除了一些文本内容。

Sharpen your pencil Solution

下面来看不同设备类的标记有什么差别，并转换为具体的代码。

❶ 对 simpler_mobile 设备类使用这个DOCTYPE。

我们将对simpler_mobile设备类使用XHTML-Basic，其他设备类都使用HTML5。

```
<?php if($device_class == 'simpler_mobile'): ?>
❶<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN" "http://
www.w3.org/TR/xhtml-basic/xhtml-basic11.dtd">
<?php else: ?>
  <!DOCTYPE html>
<?php endif; ?>
```

❷ 对 desktop、tablet 和 unsupported 设备类使用这个样式表。

应该记得，我们对unsupported设备类的处理基本上类似于desktop。

❸ 对 simpler_mobile 和 higher_mobile 设备类使用这个样式表。

❹ 对 higher_mobile 设备类使用这个样式表。

每个设备类都能得到common.css，这个文件名很贴切。

```
 <link rel="stylesheet" type="text/css" href="../assets/common.css" />
 <?php if ($device_class == 'desktop'
       || $device_class == 'tablet'
❷      || $device_class == 'unsupported'): ?>
  <link rel="stylesheet" type="text/css" href="../assets/desktop.css" />
 <?php endif; ?>
 <?php if ($device_class == 'higher_mobile'
❸      || $device_class == 'simpler_mobile'): ?>
  <link rel="stylesheet" type="text/css" href="../assets/mobile.css" />
 <?php endif; ?>
 <?php if ($device_class == 'higher_mobile'): ?>
❹ <link rel="stylesheet" type="text/css" href="../assets/mwebkit.css" />
 <?php endif; ?>
```

这些样式中的移动设备类有一些共同的元素。

这个样式表有梯度和其他漂亮的内容，专为WebKit浏览器提供。

5 对 desktop , tablet 和 unsupported 设备类显示这个版本的导航。这是桌面样式的导航，靠近页面最上方。

6 对 unsupported 设备类不要显示这两个链接。

```
<?php if ($device_class == 'desktop'
    || $device_class == 'tablet'
    || $device_class == 'unsupported'): ?>
  <div id="navigation">
    <ul>
      <li><a href="#">Home</a></li>
      <?php if ($device_class != 'unsupported'): ?>
        <li><a href="#">Shop Online</a></li>
        <li><a href="#">Study Now!</a></li>
      <?php endif; ?>
      <li><a href="#">Contact Us</a></li>
      <li><a href="#">FAQ</a></li>
    </ul>
  </div>
<?php endif; ?>
```

对这些用户，不提供在线购物或漂亮时尚的闪卡！

Flash通常不能很好地用于移动设备。平板电脑也存在同样的问题。

7 向 desktop 设备类发布Flash内容。

8 对 tablet , higher_mobile , simpler_mobile 或 unsupported 设备类不要使用Flash。

```
<?php if ($device_class == 'desktop'): ?>
  <p>Featured product Flash slideshow.</p>
<?php else: ?>
  <p>Static featured product image.</p>
<?php endif; ?>
```

继续翻到下一页。

Sharpen your pencil Solution

我们不在移动布局上显示广告，另外也不会在移动页面上放置大号的博客文字，而是提供一个指向博文的链接。

9 对 higher_mobile 或 simpler_mobile 设备类不要显示这些部分。

```
<?php if ($device_class == 'desktop'
      || $device_class == 'tablet'): ?>
  <div id="ads">
    ...
  </div>
  <div id="fromtheblog">
    ...
  </div>
<?php endif; ?>
```

这是导航的移动版本，放在页面最下面。

10 向 higher_mobile 和 simpler_mobile 设备类显示这个版本的导航。

对这些手机不提供交互式闪卡！

11 向 simpler_mobile 设备类隐藏交互式闪卡的链接。

```
<?php if ($device_class == 'higher_mobile'
      || $device_class == 'simpler_mobile'): ?>
  <div id="navigation">
    <ul>
      <li><a href="#">Home</a></li>
      <li><a href="#">Shop Online</a></li>
      <?php if ($device_class == 'higher_mobile'): ?>
        <li><a href="#">Study Now!</a></li>
      <?php endif; ?>
      <li><a href="#">Blog</a></li>
      <li><a href="#">Contact Us</a></li>
      <li><a href="#">FAQ</a></li>
    </ul>
  </div>
<?php endif; ?>
```

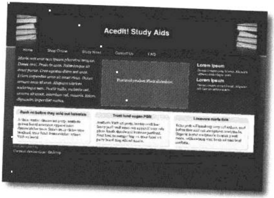
只向higher_mobile类提供闪卡的链接。

试一试

试试看!在index_solution.php中找到index.php的完成版本。把当前的index.php文件替换为index_solution.php。

在你的web浏览器中导航到test.php，尝试useful_user_agents.txt文件中的一些user-agent串，查看不同的设备类版本。可以直接用你的桌面浏览器查看这个模拟网站的桌面版本。





BRAIN BARBELL

我们改进了200页上的设备类测试，不过你还能想到可能还有哪些没有考虑到的问题?

一定要小心谨慎，做好准备

再来看第3章，我们强调了使用用户代理的服务器端设备检测存在的一些问题。有几个风险一定要记住：

1. **浏览器不一定发送正确的用户代理。**

   有时用户会有意覆盖用户代理发送的内容，有时浏览器自己就会发出奇怪的内容。这可能导致设备数据库返回不正确的数据。

2. **WURFL并不总能找到一个(特定的)匹配。**

   WURFL并不总能找到一个成功的匹配，或者可能根本找不到任何合法的匹配。你有可能只得到一个通用的设备ID，或者什么也得不到。

动手做！ → 再来看这一章之前创建的资源管理页面，输入以下用户代理：facebookexternalhit/1.1 (+http://www.facebook.com/externalhit_uatext.php)。

WURFL的“通用”太通用了

如果在我们的资源管理页面尝试这个facebookexternalhit用户代理(如果还没有，现在就试试吧)，你会注意到WURFL把它的设备ID识别为generic。通过这种方式，实际上WURFL是在耸肩摇头说“嘿，我试过了，不过我确实不知道这家伙到底是什么。”

那该怎么办？

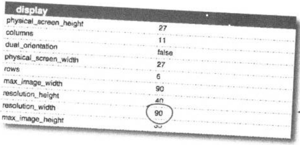
通用设备的能力值意义不大，不足以作为判断的依据。例如，facebookexternalhit客户的resolution_width不太可能正好是90像素。这只是WURFL数据文件中对应通用设备的值而已。

我们需要一个更大的安全网

设计一个使用服务器端检测的网站时，要花点时间考虑如果返回一个通用ID，网站会有什么表现，或者如果设备根本没能匹配该怎么办。提示：它应该还能正常工作。

偶尔的成功还不算成功

目前，在AcedIt!学习辅导移动测试网站上，如果用户代理匹配到一个通用设备，会接收unsupported设备类(可以用facebookexternalhit用户代理试试看)。指定为这样一个设备类应该说也算合适，不过这种分配有点靠运气。



display	
physical_screen_height	27
columns	11
dual_orientation	false
physical_screen_width	27
rows	6
max_image_width	90
resolution_height	40
resolution_width	90
max_image_height	[illegible]

管理页面的详细信息。display组的通用值过于通用，我们不能依靠这些值做判断。



Watch it!

WURFL数据可能有所不同。

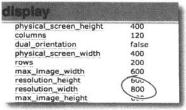
下面将我们的资源管理页面中的facebookexternalhit能力与ScientiaMobile资源管理器中的数据做个比较。

访问http://www.tera-wurfl.com/explore，并输入facebookexternalhit用户代理。首先可以注意到，Database Edition中将通用设备ID命名为generic_web_browser。

如果仔细研究返回的能力，你会发现更多差别。例如，请看显示(display)能力组，将它与以上“通用”设备ID的display能力组值进行比较。

差别很大，是不是？熟悉这种WURFL(PHP API，基于文件的版本)是很明智的，另外建立我们自己的资源管理页面确实是个好主意。

ScientiaMobile资源管理器页面的详细信息。显示了facebookexternalhit用户代理的能力。



display	
physical_screen_height	400
columns	120
dual_orientation	false
physical_screen_width	400
rows	200
max_image_width	600
resolution_height	[illegible]
resolution_width	800
max_image_height	[illegible]

及时的一针

PHP API提供了很多方法来确定匹配的特定程度。下面来看一看。

如果根本没有设备ID……

```
if (!$device->id
  || (!$device->isSpecific()
    && $device->fallBack === 'root')) {
  /* FAIL--make sure you have a plan for this situation */
}
```

或者如果匹配不特定，所以一直落到最后的根(通用)设备。



有两个原因：错误检查的好处和桌面浏览器补丁。

检查是不是缺少设备ID，这就有点类似于家务活干得好。我们要保持整洁。

通过使用桌面浏览器补丁，并结合WURFL，我们可以得到桌面浏览器以及移动浏览器的基本信息。

不过，对于所有与桌面浏览器的匹配，isSpecific()都会返回FALSE，因为一般不认为补丁中的数据是特定的。通过检查退路，并确保它不是根（通用），可以避免错误地把桌面浏览器匹配识别为失败。

there are no Dumb Questions

问：你前面谈到的桌面浏览器补丁是什么？

答： WURFL的默认安装提供了一个补丁，它能识别（范围很广的）一组桌面浏览器和移动浏览器。

问：我不太明白WURFL的哪些部分是API。

答： WURFL本身只是数据。API是与这些数据交互、组织数据等等所需的全部代码。对应多种主要的编程语言分别有多个这样的API。

问：facebookexternalhit用户代理是什么意思？

答： 有人在Facebook上分享一个链接之类的东东时，Facebook通常会从那个页面去获取一些内容，而且/或者会给出一个代表性的缩略图。这时它用的用户代理就是facebookexternalhit。

问：设备类测试代码作为一个函数/对象/或者采用其他形式不是更好吗？

答： 确实。事实上，对于真正的项目，代码可能会更精巧，更高雅，更有条理，而且更强大。在这里我们只是向你展示基础的知识。

问：抱歉，不过如果我没有想错，$device变量是一个全局变量。这太糟糕了。

答： 看看上面的介绍。

Exercise

更新adapt_content目录中的device_classes.php，使它包含测试通用ID或查找失败的情况。这种情况下要指定为unsupported设备类。

提示：代码要放在这里。

```
$device_class = NULL;

if (device_match('is_wireless_device', '===', 'false')) {
  $device_class = 'desktop';
}
```

device_classes.php

Exercise Solution

现在我们已经攻克了所有难关！大功告成！

```
$device_class = NULL;
if (!$device->id ||
    (!$device->isSpecific() && $device->fallBack === 'root')) {
  $device_class = 'unsupported';
}
else if (device_match('is_wireless_device', '===', 'false')) {
  $device_class = 'desktop';
}
```

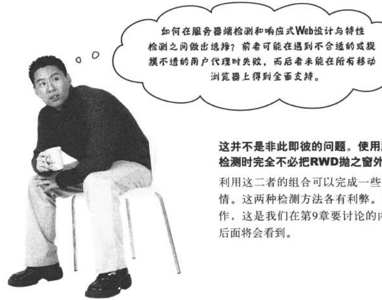
device_classes.php



祝贺你，你已经掌握了难缠的设备能力和设备类。

这是一个复杂的概念，不过你做到了！甚至还完成了一个适应内容调整的Web页面作为证明。干得不错！





这并不是非此即彼的问题。使用服务器端设备检测时完全不必把RWD抛之窗外。

利用这二者的组合可以完成一些非常漂亮的事情。这两种检测方法各有利弊。让二者协调合作，这是我们在第9章要讨论的内容的一部分，后面将会看到。

BULLET POINTS

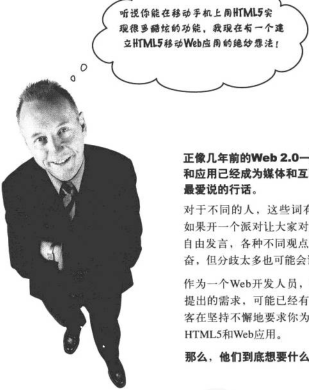
- 利用设备数据存储库（如WURFL，代表无线通用资源文件，Wireless Universal Resource FiLe），我们可以得到大量设备的非常详细的信息。
- WURFL数据包含大量信息，每个设备超过500个能力，这些设备组织为二十多个组。
- 可以使用一个设备数据库找出一个给定能力的值，并根据这个值采取相应的行动。
- 与WURFL数据交互的API有很多，WURFL PHP API是其中之一。不同的API处理和表示数据的方式可能稍有不同。
- ScientiaMobile是2011年由原先的一些WURFL维护者创立。这家公司提供了WURFL的开源和商业许可证。如果WURFL不适用，还有其他一些选择。
- 处理较大的项目时，将相关的能力分组为设备类会很有帮助。
- 设备类是根据共同能力建立的抽象的设备分组。
- 通过将一个设备分组到一个设备类，我们可以对它采取进一步的处理（例如，发布适应性的内容），而不必一直跟踪单个的能力。
- 如果代码中要使用一个设备数据库识别设备，建立默认设备类、错误检查和通用设备处理会很重要。
- 与Web上的所有一切类似，服务器端设备检测不是一个100%可靠的概念。
- 服务器端检测和内容适应性调整可以与客户端适应性调整相结合，这二者不是互斥的。后面将介绍更多有关内容。

6 使用框架构建移动Web应用

花格布爱好者



“我们想要个应用！” 最近一到两年，我们常常听到这样的呼吁，这通常意味着要为你想要支持的每一个平台完成原生代码开发和部署。不过，原生应用不再独占江山。如今，面向移动浏览器的Web应用已经蜂拥而至，特别是现在HTML5和他的死党CSS3和JavaScript都在摩拳擦掌、跃跃欲试。下面就让我们利用一个移动框架更深入地探寻这个移动Web应用世界，移动框架就是专门设计用来帮助你快速完成工作的代码工具！



正像几年前的Web 2.0一样，HTML5和应用已经成为媒体和互联网领域人们最爱说的行话。

对于不同的人，这些词有不同的含义，如果开一个派对让大家对这些词的理解自由发言，各种不同观点可能很令人兴奋，但分歧太多也可能会让人困惑。

作为一个Web开发人员，你要面对用户提出的需求，可能已经有一些性急的顾客在坚持不懈地要求你为他们专门建立HTML5和Web应用。

那么，他们到底想要什么？

HTML5……应用……这些词究竟是什么意思？

HTML5很明确……

HTML5很明确。这是一个正在发展的标准，是我们了解和喜爱的HTML的一个演进版本（如果没有这种语言，根本不会有现在的Web）。HTML已经有20多年的发展历史，HTML5在这个标记语言基础上做了进一步调整和改进，特别是增加了对Web应用的支持，而不会对向后兼容性有太多影响。

HTML5引入了一些新的语义元素，如<section>、<article>、<nav>和<header>。它简化了一些标记的语法，利用<audio>和<video>标记为我们提供了强大的媒体能力，另外通过一些新的JavaScript API（如地理定位和离线存储）大大解除了对交互性的束缚。

HTML

……不过还有更多

人们谈到HTML5时，通常是指HTML5本身、超炫的JavaScript、还有漂亮的CSS3这三者的组合。简单地说，这些正是构建现代交互式Web应用的核心部分。

有时它们就表示“我在Web上看到的最酷的东西”。

嗯，这有点容易让人误导。

……那么，到底什么是Web应用？

HTML5 Web应用中的“Web应用”部分到底是什么？HTML5本身已经很令人困惑了，再结合“应用”这个词，情况会更糟糕，你将进入一个更昏暗的充满危险和分歧的世界。

你可能经常在浏览器中查看网站，根据你看到的网站表现来体会这是否像是一个应用。要强调任务的完成，布局要占满全屏，另外动作不会导致整个页面重新加载，还有交互性，所有这些都可以作为准则来判断这是一个应用而不只是以内容为中心的常规网站。

简单地说，任何人都无法对“应用”这个词给出一个让所有人都满意的完美定义。

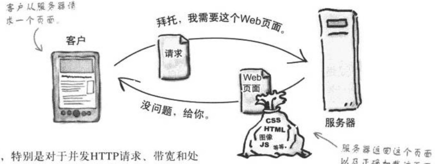
Web应用很难定义，不过它们都具有一些交互性特点，非常适合发挥HTML5及相关技术的强大威力。

"传统"网站的一般表现

在传统的网站模型中，一个HTML页面请求会获取这个页面的所有组件，包括HTML本身、JavaScript、CSS、图像等。

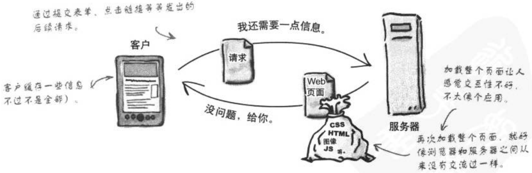
浏览器通常很擅长缓存元素，另外可以把Web服务器配置为鼓励更多缓存。不过这个模型意味着每次交互（包括提交表单、点击链接等等）都会导致加载完整的页面。

第一次页面请求



这有一个缺点，特别是对于并发HTTP请求、带宽和处理能力很有限的移动设备来说，这意味着会请求和下载浏览器之前已经下载过的资源，而且会因为重新生成内容和重新处理可能没有任何改变的代码而影响浏览器的正常工作。另外交互性也不太好。

后续请求



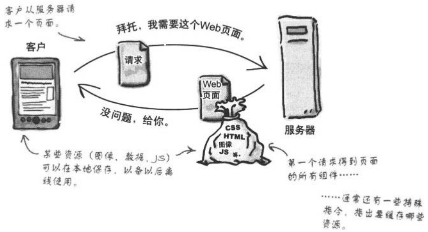
应用型网站的一般表现

在应用型模型中，客户会扮演更重要的角色，每个请求中传递的资源会更少。可重用的标记、代码和资源可以在本地存储。另外使用Ajax异步请求已修改的内容或数据。这些后台异步请求会“拉入”特定的内容或资源，而不会导致重新加载整个页面。

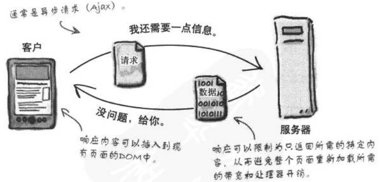
只请求有关的部分内容而不是重新加载整个页面，这样可以减少带宽和处理负担，改善交互性。

很多交互性都是通过异步请求完成的。

第一次页面请求



后续请求



Tartans Unlimited的移动HTML5 Web应用

Tartans Unlimited是一家国际机构，正努力传承苏格兰花格布的相关历史和文化传统，让它在世界其他地方得以发扬光大。





Jill：嘿，朋友们，我知道需求还很含糊。和Ewan交谈时，我有种感觉，他希望这个网站（噢，按他的说法，也可以说是应用）主要完成两件事。现在有一大堆内容页面，不仅有关于这个组织的信息，还有关于花格布的内容。另外还有整个这个板块，他把它叫作Tartanator（花格布爱好者）。

Joe：Tartanator？不是开玩笑吧？哈。

Jill：是的，千真万确，实际上他希望整个网站就叫这个名字，而不只是一个版块。毕竟，看来这是一个组合，不仅要提供一种可浏览的方式列出花格布图案供用户研究，另外他还希望能够让用户利用一个表单创建自己的花格布图案。

Frank：哇，听上去很震撼，让人很畏惧，不过作为一个挑战，实现起来可能会很有意思。

Joe：OK，内容页面，Tartanator板块，这些我知道了……那么，关于“事件”页面的这个说明又是什么意思？

Jill：别着急！我就要谈到这个问题了！我们已经决定通过两个阶段完成这个项目。第一阶段中，我们会建立内容页面的基本结构，并实现花格布列表。他还希望我们考虑为用户提供一个界面来创建他们自己的花格布，他想知道这个界面是什么样的，可能还希望我们提供这个界面的一个原型。

Joe：嗯，好吧，我们会建立创建花格布的表单，不过目前它不需要做具体工作，是吗？

Jill：对，差不多吧。在第2阶段，我们会让它真正开始工作，那时还会回来处理这个“事件”板块。

Frank：OK，看来现在可以开始构建一个移动Web应用了。



Tartanator第1阶段主计划

☐ **构建内容页面和网站结构。**

需要创建基本版块和页面，并创建一个整体结构。

这是我们要做的！

☐ **创建花格布列表。**

对于第一阶段，我们将创建一个包含现有流行花格布图案的列表。花格布版块应当是一个浏览界面，当然，感觉要像个应用，而且要面向移动设备。

☐ **构建花格布创作表单的一个原型。**

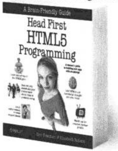
最后，Ewan希望用户能够使用一个类似应用的移动界面创作他们自己的花格布。他想看看这个界面是什么样子，所以我们要为他设计一个原型。

这么说，我们要用HTML5和那些相关技术构建一个移动Web应用。我真不知道从哪里下手。所有这些都得从头开始构建吗？



嗯，你可以从头开始构建一个应用……

如果你想掌握CSS、HTML和JavaScript，了解如何让它们合作创建一流的Web应用，可以看看这本书《Head First HTML5 Programming（中文版）》。



买下吧，这本书非常棒！这是市场上最棒的HTML5编程书。说真的，你知道你要做什么。

好了，小编，别插话了！

PDG

……或者你可以使用一个移动Web框架

要构建Tartanator网站的第一阶段，可以使用一个移动Web框架，迅速提高我们的Web水平。

仍然要使用HTML和CSS，不过面向移动的用户界面框架可以帮助我们更快地完成工作。

为什么使用移动Web应用框架？

面对现实吧。要想从头开始构建复杂的交互式Web应用，这可能是一个让人望而生畏的工作。Web开发框架是一组打包的交互式元素和代码工具，可以为我们助一臂之力。

1. **框架可以帮我们让网站或应用看上去是移动友好的。**

   面向移动（Mobile-oriented）的框架一般能帮助改变和设置HTML元素的样式，使它感觉更面向移动，这通常可以节省我们大量时间。

2. **框架可以帮我们让网站或应用感觉是面向移动的。**

   可以由框架来完成过渡和效果之类的繁杂工作，让网站或应用感觉更自然（或者至少更一致）。

3. **框架可以帮我们管理跨平台不一致性。**

   框架开发人员能充分驾驭那些讨厌、奇怪的浏览器特异问题，否则它们可能会带来很大麻烦，框架开发人员会在框架的代码基中解决这些问题，使我们免遭此劫。

根据具体的项目，你可能不希望一切都重新开始，起码不希望从头开始创建交互式Web应用所需的某些部分。在这种情况下，移动开发框架就有了用武之地。

Watch it!

要小心选择和使用Web框架。

框架是非常强大的东西，不过它们也有一些弊端。很多框架非常庞大，可能会让你的网站增加数百KB，使网站膨胀。有些人会采用一种厨房水池的方法，在网站中加入大量部件和花哨的动画，这样不仅会让网站的字节数大幅增加，在能力较弱的设备上还可能导致严重的性能问题。最后一点，一定要对框架的设备支持情况进行评估。有些框架只支持一个或几个主要的前沿平台。

Tartanator的选择：jQuery Mobile

要构建这个Tartanator网站，我们将使用jQuery Mobile框架。

jQuery Mobile是一个专门为移动设备优化设计的用户界面框架。它建立在相当流行的jQuery JavaScript库基础上。

我们选择jQuery Mobile完成这个项目的原因是，这个框架很容易使用，而且它的体系结构可以很好地映射到我们的HTML5中心，jQuery Mobile采用一种巧妙的方式设计，可以轻松地（有时甚至是不可见地）关联到良构的HTML5标记。

另外，如果你以前用过jQuery，就会知道它是多么直观和简单。

jQuery mobile framework.

与所有jQuery项目一样，jQuery Mobile也是开源的。

我们已经掌握了很多不同的方式来构建移动Web内容。交互式Web设计、单独的网站、使用设备检测或特性检测。我是不是得把所有这些都忘掉，再使用我们的移动Web框架从头开始？



你说的没错。这里情况确实很复杂。

正是因为这一点，我们才把移动Web世界比作原先蛮荒的西部世界。不过，这确实不是危言耸听。这里没有简单的答案，在移动Web领域，提出复杂的解决方案通常需要结合多种实现方法。

所以，你不能（也千万不要）丢掉我们已经掌握的知识。到目前为止我们介绍的所有技术都不是互斥的，在不同环境下，每一种技术都或多或少有某种程度的价值。

生活很复杂。移动Web也是如此！

there are no Dumb Questions

问：还有没有其他移动框架？

答： 当然了！每天都有更多移动框架问世。其他移动web框架包括Sencha Touch、Wink、iUI、DHTMLX Touch和JQTouch。

问：移动Web框架到底由什么组成？

答： 这取决于具体的框架，不过大多数都是JavaScript、CSS和图像（或其他）资源的组合，用来帮助设置样式，丰富移动体验。有些框架还包括一个服务器端组件，可以帮助生成（而不是调整）内容。

问：等一下，zepto.js或XUI呢？

答： Zepto.js（非常轻量级的JavaScript，使用jQuery语法）和XUI（也是非常精简的JS）与其说是框架，不如说是库。框架通常有UI组件，而库往往只是代码（在这里就是JavaScript代码）。这是一个灰色地带；库与框架的划分不容易定义。

问：这么说，jQuery Mobile就是原先jQuery库的一个移动版本吗？

答： 别那么快下结论，真性急！jQuery Mobiie建立在jQuery基础之上，但没有取代它。你会注意到，我们用jQuery Mobile构建网站时，包含的第一个JavaScript文件就是核心jQuery库。

问：所以，jQuery Mobile是一个扩展jQuery的JavaScript开发框架。

答： 耐心一点。没错，jQuery Mobile框架里确实有JavaScript。不过，jQuery Mobile并不是一个JavaScript框架。它比这范围更大。这是一个用户界面框架。这说明，它还包括样式表、图标和其他一些东西。

问：我们真的需要使用框架吗？

答： 需要吗？理论上需要吗？不，根本不需要！实际上，建议你完全可以从头构建一个应用型的移动网站（如果你办得到）。

不过，框架（尤其是jQuery Mobile）有一点好处，它们能为我们解决很多烦人的特定于平台的奇怪问题和bug。框架开发小组对不同移动浏览器的各种怪问题很有研究。

我们的时间和篇幅都有限。如果没有某个框架的帮助，要想实现这个Tartanator网站所需的功能，可能相当棘手，更不用说可能还需要大量额外的测试（因为我们没有经过充分测试的跨平台支持作为基础）。

问：我还是不明白。为什么这个Tartanator是一个应用而不是网站？

答： 因为Ewan这么说。不，开玩笑的。应用和网站之间的细微差别很模糊，所以答案有时几乎是随意的。

对Ewan来说，这个Tartanator网站就是一个功能丰富的Web应用。他的关注点在于用户能够查找和创建花格布，另外，最后还可以搜索相关的事件。在他看来，这些可以说明它就是一个应用。

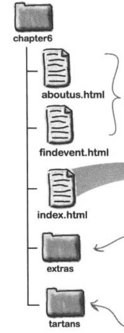
问：但是如果它是网站呢？我就不能使用jQuery Mobile了，是吗？

答： jQuery Mobile是一个用户界面框架。它并不关心你要构建的东西叫作应用还是叫作网站。它的任务就是使用CSS、JavaScript和HTML5组合，让你构建的应用（网站）可用，在不同的移动平台上不会出问题。

问：那么，哪些移动浏览器支持HTML5？

答： 没有哪一个浏览器支持（全部）HTML5。别担心！HTML5是模块化的，移动浏览器已经开始支持越来越多的HTML5模块。实际上，任何现代桌面浏览器都没有全面地支持CSS2.1的每一个部分，类似地（正如规范中所指出的），HTML5未得到所有浏览器的全面支持也是可以理解的。噢，这个规范还在发展。网站http://www.caniuse.com是一个很好的参考，如果你要查找HTML5和其他主要Web技术中某些特性支持情况的有关信息，可以访问这个网站。

用jQuery Mobile建立一个基本页面



chapter6目录结构：可以作为你的起点

Tartanator之旅的第一站是学习如何用jQuery Mobile建立一个简单的页面，这样我们就可以开始建立这个项目中的页面。你会惊奇地发现这是多么的容易。

从最基本的开始

先从一个超级简单的HTML页面开始，这样你可以更清楚地了解具体使用jQuery Mobile的一些基础知识。

再过几页我们还会见到这些页面。

我们使用了HTML5 <DOCTYPE>标记。

这就是index.html的全部内容，说真的，就是这么简单！

现在可以先忽略这个文件夹。稍后我们会告诉你里面是什么。

这个文件夹也稍后再介绍。

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8" />
    <title>The Tartanator</title>
</head>
<body>
<h1>The Tartanator</h1>
<p>The Tartanator is a community-built association of
groups, businesses, and individuals bent on keeping
the Scottish heritage alive overseas by promoting the
understanding and enjoyment of <strong>tartans</
strong></p>.
</body>
</html>
```

index.html

包含jQuery移动代码组件

在文本编辑器中打开index.html，准备加入一些jQuery Mobile组件。最首要的是，要包含基本的JavaScript和CSS。

通过包含jQuery和jQuery Mobile JavaScript，这个Web应用将以移动友好的方式提供交互。

包含jQuery Mobile的CSS：这会以移动友好的方式调整标记的样式。

要记住，如果没有jQuery（核心），jQuery Mobile什么也不是。

```
<title>The Tartanator</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="http://code.jquery.com/mobile/1.0rc1/
jquery.mobile-1.0rc1.min.css" />
<script src="http://code.jquery.com/jquery-1.6.4.min.js"></script>
<script src="http://code.jquery.com/mobile/1.0rc1/jquery.mobile-1.0rc1
min.js"></script>
</head>
```

index.html

页面其余的标记

我们已经从jQuery的CDN（内容分发网络，即在code.jquery.com托管的代码）包含了3个主要jQuery Mobile组件。这是一个可以快速链接代码的可靠的位置。

我们已经包含了3个核心文件—主题CSS、jQuery内核和jQuery Mobile本身，现在需要对<body>中的HTML标记做一些调整。

我们把各部分内容包围在<div>标记中，并为这些<div>标记指定描述性的data-*属性，通过这种方式告诉jQuery Mobile我们希望如何处理这些内容：

稍后就会更详细地解释data-*属性。

data-role为"page"，这是在告诉jQuery Mobile要把这个内容处理为一个完整的页面。

data-role为"header"则告诉jQuery Mobile要调整这个内容的样式，使它成为一个标题元素。

data-role为"content"表示（应该能猜得到）这是页面的主要内容。

下面再加上一个页脚，看看jQuery Mobile对它如何处理。

```
<body>

<div data-role="page">
  <div data-role="header">
    <h1>The Tartanator</h1>
  </div><!-- /header -->
  <div data-role="content">
    <p>The Tartanator is a community-built
association of groups, businesses and individuals
bent on keeping the Scottish heritage alive
overseas by promoting the understanding and
enjoyment of <strong>tartans</strong>.</p>
  </div><!-- /content -->
<div data-role="footer">
  <h4>Bring forrit the tartan!</h4>
</div><!-- /footer -->
</div><!--/page-->
</body>
```

index.html

"Bring forrit the tartan!"是一句令人振奋的战斗口号，这是19世纪50年代甘柯林爵士在《siege of Lucknow》一书中提出的。



试一试

建立一个简单的jQuery Mobile页面确实这么容易。

在index.html文件中增加首部和主体元素，并保存。在浏览器中查看（如果愿意，你也可以在一个移动浏览器中查看）。

我们动身吧！

目前还没有太大意思（嗯，确实很乏味），不过这个登陆页面看起来确实有点“移动”味儿了。

注意jQuery Mobile如何增加梯度、大小和字体处理，以及如何把我们的元素转变成类似页眉和页脚。

我们的第一个jQuery Mobile页面（在iPhone上显示的结果）。



嗯……页脚文件不太合适……不过稍后再考虑这个问题。



你说“看起来是移动的”到底什么意思？难道这个页面不更像一个原生的iOS应用吗？

默认的jQuery Mobile看起来绝对是iOS范儿。

毕竟，圆角按钮、梯度颜色绘制的标题这些原生iOS平台的外观已经成为移动Web应用的常见标志性象征。很多人现在都会把这种外观与“移动”关联起来。

不过除了Apple式的美化，这里还有一些有用的东西。为便于小屏幕阅读，这里增加了空白符，另外优化了字体大小。稍后我们还会构建表单，到时会看到jQM为我们提供了更大的输入区和更大的按钮（省得我们的胖指头按错键）。

这还只是几个例子！

HTML5 data-*属性

jQuery Mobile大量使用了HTML5新引入的data-*属性。在这个简单的页面中，我们使用了data-role属性告诉jQuery Mobile给定元素在这个页面结构中的角色。对现在来讲，情况很简单：我们有一个页眉，一些内容，还有一个页脚。

为Tartanator构建更多页面

我们已经有了一个坚实的基础，下面为Tartanator构建更多页面。现在已经有了登录页面，不过还需要让它链接到另外一些基本页面，继续推进我们的项目。

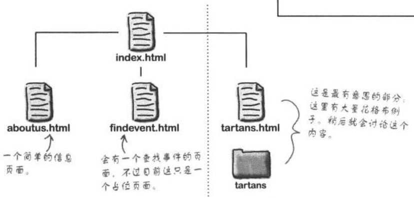
HTML5 data-* 特写

data-*属性是HTML5引入的，利用这些属性，开发人员可以为HTML元素关联很有意义的轻量级数据。

过去，为了达到类似这样的目的，开发人员也尝试过一些方法，比如通过"劫持"另外一些元素属性（如class或title）来实现。

不过，现在已经有了真正的、更有实效的方法，jQuery Mobile充分利用了这一点。

在我们的前进道路上还会看到，jQuery Mobile会使用不同的data-*属性来完成任务。



Exercise

现在增加一个类似导航的链接列表，链接到这3个子页面。编辑index.html，在主内容<div>中的介绍段落后面增加一个非常简单的<ul>。列表元素的文本应该就是子页面的名字：分别是About Us、Find an Event和Popular Tartans。

稍后我们会增加具体的链接，不过现在它们只是文本，OK？

Exercise Solution

基本HTML FTW！

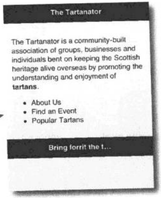
```
<p>The Tartanator is a community-built association
of groups, businesses, and individuals bent on keeping
the Scottish heritage alive overseas by promoting the
understanding and enjoyment of <strong>tartans</strong>.</p>
  <ul>
    <li>About Us</li>
    <li>Find an Event</li>
    <li>Popular Tartans</li>
  </ul>
</div><!-- /content -->
```

index.html

建立jQuery Mobile列表

如果保存你的修改，并在浏览器中查看index.html，可能会注意到这个小列表看起来并不让人满意（感觉没有"移动性"）。这是因为我们还没有通知jQuery Mobile希望它注意这个列表的存在。这里需要再次使用data-role属性！

嗯，这是个很普通的列表。




```
<ul data-role="listview">
```

我们使用data-role属性告诉jQuery Mobile，要把它处理为一个列表视图。

这样一来，jQuery Mobile就会相应地设置这个列表的样式，并注意它的存在。

jQuery Mobile会修改data-role为listview的内容，使它看起来是一个移动友好、更有交互性的列表。

试一试

在index.html中为<ul>增加data-role属性，再在浏览器中查看这个页面。

你认为jQuery Mobile会如何设置这个列表的样式？

我们的列表好多了，但还不完美

默认地，jQuery Mobile会把listview当作页面内容来处理，<li>元素会占满整个屏幕宽度。在我们的登录页面上，这看起来不美观，还很压抑。

对我们来说，导航列表是内容的一部分，而不是全部。jQuery Mobile还有一种列表选择，即内插列表，也就是包含这些列表的页面上还有其他内容。下面就使用这种内插列表。

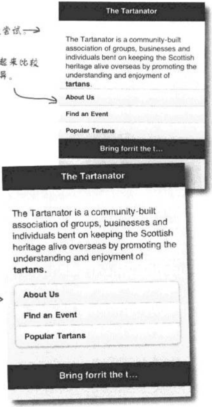
```
<ul data-role="listview" data-inset="true">
```

第一次尝试。

这个列表看起来比较挤，有些怪异。

第二次尝试。

内插列表看起来好多了。



动手做！

向index.html中的列表增加data-inset属性，再试着查看这个页面。

这个列表看起来很不错，不过所有元素都不可点击。嗯，难道不需要让它们真正链接到什么地方吗？



你说的对。下面就来链接这些页面！

现在要为Tartanator多加几个页面，而不再只是一个页面演独角戏了。

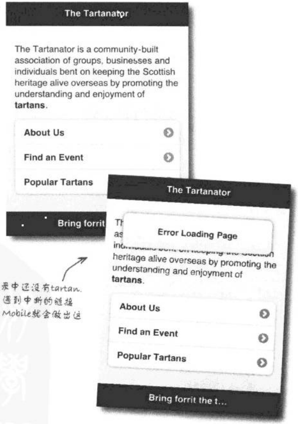
用jQuery Mobile链接到多个页面

jQuery Mobile中要链接到其他页面相当简单。我们要做的就是增加基本的HTML链接：

```
<ul data-role="listview" data-inset="true">
  <li><a href="aboutus.html">About Us</a></li>
  <li><a href="findevent.html">Find an Event</a></li>
  <li><a href="tartans.html">Learn about Tartans</a></li>
</ul>
```

index.html

动手做！ 把这些链接增加到index.html中的列表，然后做个测试。会发生什么？



通过增加链接，我们会自动得到这些箭头图标，告诉用户这些是可点击的链接。

我们的主目录中还没有tartan.html文件。遇到中断的链接时，jQuery Mobile就会做出这种反应。



jQuery Mobile闪亮登场

本周访谈：

jQuery Mobile如何处理登录页面？

Head First：我听说你有话要对我们说，要告诉我们“页面”对你来说到底是什么。

jQuery Mobile：没错！这是我对自己最满意的方面之一，所以我想解释一下。

还记得吧？之前一个页面的结构使用了一个<div>，而且data-role值设置为page。

Head First：当然记得。

jQuery Mobile：包含在那个<div>中的标记（通常包括一个页眉、内容区和页脚）在我看来就是一个页面。

Head First：那么其余的HTML标记做什么用呢？只是些边角料吗？

jQuery Mobile：当然不是！在按我的原则建立的web应用中，每个HTML文件都是一个独立的、自治的页面，都可以在浏览器中访问。

Head First：我不明白。如果每个HTML文件都是独立的，那又何必还要有data-role为page的<div>呢？

jQuery Mobile：先退一步讲。你第一次导航到我的网站或应用的一个页面时，这个页面会像Web上的所有传统HTML页面一样进行加载。

但是，一旦加载了第一个页面，我就有所不同了。你点击链接时，我会从所请求的HTML文档找到页面内容，也就是data-role为page的<div>中的内容，再利用AJAX把它插入到当前页面的DOM中。

Head First：不过这有什么意义呢？

jQuery Mobile：当然有意义，如果你听我说完！

我不再请求和下载整个页面（包括脚本、图像、样式），然后再从头开始重新初始化并重建DOM，现在我只需要挑出我关心的部分。这样可以节省HTTP请求、带宽和处理时间，让用户得到更自然的体验。

Head First：如果内容部分是动态加载的，我怎么在桌面浏览器的地址栏上看到我点击的那个链接的的URL呢？

jQuery Mobile：尽管我的导航模型外表看起来很朴实，实际上它城府很深。我一直都在努力为应用中的页面提供真正的URL，即使页面实际上是动态生成的，或者一个HTML文档中有多个data-role为page的<div>标记。

通过为Ajax加载的内容块提供一个唯一、可重用的URL，我可以让它们看起来就像一个完整的Web页面。浏览器允许时（要知道不是所有浏览器都允许），我甚至还可以在异步加载新内容时更新浏览器地址栏中显示的URL。

Head First：如果我理解得对，这么说，我可以直接访问应用或网站中的各个HTML文件，不过链接时这些文档的内容可以独立地动态获取，这样可以改善性能和响应性，是吗？

jQuery Mobile：完全正确！

Head First：非常感谢，jQuery Mobile，与你交谈真让我们收获不小。



即使根本没有JavaScript支持，jQuery Mobile也能正常工作。 不可能吧？

还记得吧，标记中的链接刚开始只是“基本的HTML链接”。是jQuery Mobile的JavaScript施了魔法，对于那些支持JavaScript的浏览器，jQuery Mobile会把这些链接转换为动态的AJAX式链接。

而对于不支持JavaScript的浏览器，页面之间的导航还是采用老式的Web导航，所以仍然可以正常工作。

there are no Dumb Questions

问：如果用Ajax加载页面，为什么还需要创建单独的页面呢？难道不能把应用的所有内容都放在一个页面上，再使用jQuery Mobile来显示和隐藏吗？

答：当然也可以，不过这样一来，你会丢掉jQuery Mobile的一个很好的特性：渐进增强。那些不支持高级JavaScript、能力不太强的移动浏览器就会被冷落到一边。

另外，这个页面会变成一个庞大、复杂的文件，这个文件不仅很难维护，而且有一个笨重的DOM，对于能力较弱的手机来说将很难处理。jQuery Mobile鼓励你像以往那样创建网站和应用，即创建单独的HTML，它们可以独立存在，也可以用Ajax加载。

问：jQuery Mobile怎么知道要用哪个data-*属性？在哪里可以找到所有可用data-*属性的信息？

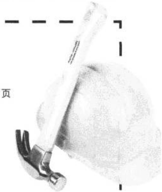
答：开发人员可以为data-*属性构造他们喜欢的任何名字（嗯，必须以字母开头）。对此没有相关规定。重点是，data-*属性要与它们所在网站的功能有关。也就是说，它们不用于向外部应用传递数据。由于这个原因，Web开发人员可以充分自由地生成他们自己的data-*命名约定。

问：我们在使用jQuery Mobile的哪个版本？

答：jQuery Mobile开发小组真是很卖力！在我们写这一章这短短的时间里，就已经发布了1.0 beta 3和1.0 Release Candidate 1版本，我们又得奋力赶上。

噢，等一等，我们刚查过。1.0RC2现在已经过气了！噢，RC3也是！哎呀，这简直比子弹还飞得快！现在已经官方发布了jQuery Mobile 1.0！

对于Tartanator，我们会使用1.0RC1。



页脚（重）构

在230页上已经看到，在窄屏幕上页脚中的文本被截断了。

这是因为jQuery Mobile要在页眉和页脚中为标题元素周围放置按钮留出空间，不过我们的页脚中没有按钮。现在来修正这个问题！

重构前：我们的页脚被截断了！

另外，在视觉上有点过于突出了。感觉太暗。



默认地，jQuery Mobile会为页眉和页脚中的标题元素（<h1>等）周围放置按钮留出空间。

1 删除包围页脚文本的<h4>元素。

现在页脚文本包围在一个<h4>元素中。通过去除这个标题元素，就可以不让jQuery Mobile为按钮留出空间（为按钮留出空间还有一个副作用：会截断文本）。

```
<div data-role="footer">
  <h4>Bring forrit the tartan!</h4>
</div><!-- /footer -->
```

index.html

2 增加一个CSS规则，将页脚文本居中，并指定页边距。

遗憾的是，删除<h4>也会同时删除居中设置和页边距。需要在我们自己的CSS中考虑这个问题。

在文本编辑器中打开tartans/tartans.css，把以下规则增加到这个文件的最上面（文件中已经有很多CSS规则）。

```
[data-role="footer"] { text-align: center; padding: 5px 0;}
#abercrombie { background-image:url('icons/abercrombie.png'); }
```

这个CSS选择器会应用到data-role属性值为"footer"的元素。

tartans.css

继续翻到下一页

页脚（重）构

3 链接样式表。

再来编辑index.html，增加一个样式表链接。

```
<link rel="stylesheet" href="http://code.jquery.com/mobile/1.0rc1/
jquery.mobile-1.0rc1.min.css" />
<link rel="stylesheet" href="tartans/tartans.css" />
<script src="http://code.jquery.com/jquery-1.6.4.min.js"></script>
```

index.html

4 检查我们的进度。

保存文件，在一个智能手机浏览器或模拟器中查看index.html。

页脚不再截断。不过还是感觉太暗。另外，看呐，它并不总在屏幕的最下方出现。这里还有一道缝隙。这让它感觉有点怪异。肯定有办法修正这个问题……



要修正这两个问题，需要对页脚<div>做两个小小的修改。

我们想告诉jQuery Mobile使用固定定位让页脚固定出现在页面最下方，另外需要改变主题swatch，使页脚在视觉上不那么突出。



页脚并不总在页面的最下方出现，有时这里还有一道缝隙。



5 **在页脚上使用固定定位，使它总出现在相同的位置上。**

使用data-position属性，并设置它的值为fixed，这就是在告诉jQuery Mobile对页脚元素使用固定定位。这样一来，它就总能固定出现在页面的最下方。

6 **使用一个不同的主题swatch，使页脚不那么突出。**

jQuery Mobile的初始样式表有5个默认的颜色组，这称为swatch。这5个swatch分别由字母a到e引用。

默认地，页眉和页脚元素会采用swatch a，这种反差最突出。

我们的页脚并不是一个特别重要的页面元素，目前它的视觉反差太大了。通过指定为swatch c，可以让页脚看起来不那么突兀。

可以在任何元素上使用data-theme属性来覆盖jQuery为该元素设置的默认swatch。

swatch a到e的默认颜色样式在jQuery Mobile的CSS中定义。

```
<div data-role="footer" data-position="fixed" data-theme="c">
  Bring forrit the tartan!
</div><!-- /footer -->
```

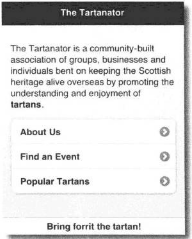
index.html

就这么简单！现在页脚应该看上去好多了！保存所做的修改，再在一个移动浏览器或模拟器中查看改进后的页脚。

jQuery Mobile 1.0包括一个新的Theme Roller工具，使主题jQM使用更为容易，有关的更多内容参见http://jquerymobile.com/themeroller/。

Tartanator的核心：花格布

下面来看现在已经完成到Tartanator项目第1阶段的哪一步：



现在页脚的位置是正确的，视觉上也不那种突出。成功了！

☑ **构建内容页面和网站结构。**

需要创建基本板块和页面，并创建一个整体结构。

我们已经建立了基本布局，并创建了主要页面的基本内容。

☐ **创建花格布列表。**

对于第一阶段，我们将创建一个包含现有流行花格布图案的列表。花格布板块应当是一个浏览界面，当然，感觉要像个应用，而且要面向移动设备。

下一个任务！

☐ **构建花格布创作表单的一个原型。**

稍后就会完成这个任务。

我们已经有了核心的内容页面（至少有了这些页面的骨架）。现在要转移视线，来关注更有意思的部分：花格布本身。

对于第1阶段，Tartanator允许用户浏览一组流行的花格布图案。可以认为这就像是一个快速的花格布参考。

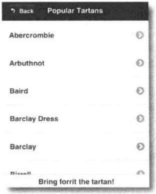
动手做！ 把tartans.html从extras文件夹复制到chapter6文件夹（也就是说，把它上移一层）。

你的chapter6目录中已经有一个tartans目录（包含各个tartan HTML页面）。

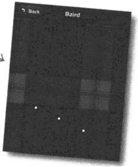
处理tartans.html

tartans.html文件已经有一个初始的流行花格布列表(<ul>)。为了便于你起步，这个列表包含了名字以A和B开头的花格布。一旦把tartans.html文件移到合适的位置，在浏览器（移动或桌面浏览器）上加载Tartanator，查看这个页面。可以点击这些花格布名字，访问相应的花格布页面。

列表领先一步



tartans.html



tartans/baird.html

让列表麻雀变凤凰

与花格布信息页面本身相比，花格布列表看起来太平庸了。好消息！利用jQuery Mobile，你可以很轻松地在列表中加入缩略图，这样列表本身可以有小图标（这就漂亮多了）。下面给出一个例子：

没错，就这么简单。你要做的只是增加一个图像，jQuery Mobile会负责其余的事情。

```
<li><a href="tartans/abercrombie.html">
  <img src="tartans/icons/abercrombie.png" alt="Abercrombie" />
  <h3>Abercrombie</h3>
</a></li>
```

Exercise

在tartans.html中为列表项增加缩略图。图标放在tartans/icons目录中，与其HTML文件同名（不过扩展名是.png）。

Exercise Solution

```
<ul data-role="listview">
  <li><a href="tartans/abercrombie.html">
    <img src="tartans/icons/abercrombie.png" alt="Abercrombie" />
    <h3>Abercrombie</h3>
  </a></li>
  <li><a href="tartans/arbuthnot.html">
    <img src="tartans/icons/arbuthnot.png" alt="Arbuthnot" />
    <h3>Arbuthnot</h3>
  </a></li>
  <li><a href="tartans/baird.html">
    <img src="tartans/icons/baird.png" alt="Baird" />
    <h3>Baird</h3>
  </a></li>
  <li><a href="tartans/barclay-dress.html">
    <img src="tartans/icons/barclay-dress.png" alt="Barclay Dress" />
    <h3>Barclay Dress</h3>
  </a></li>
  <li><a href="tartans/barclay.html">
    <img src="tartans/icons/barclay.png" alt="Barclay" />
    <h3>Barclay</h3>
  </a></li>
  <li><a href="tartans/birrell.html">
    <img src="tartans/icons/birrell.png" alt="Birrell" />
    <h3>Birrell</h3>
  </a></li>
  <li><a href="tartans/blair.html">
    <img src="tartans/icons/blair.png" alt="Blair" />
        <h3>Blair</h3>
  </a></li>
  <li><a href="tartans/borthwick-dress.html">
    <img src="tartans/icons/borthwick-dress.png" alt="Borthwick Dress" />
      <h3>Borthwick Dress</h3>
  </a></li>
  <li><a href="tartans/borthwick.html">
    <img src="tartans/icons/borthwick.png" alt="Borthwick" />
    <h3>Borthwick</h3>
  </a></li>
    <li><a href="tartans/bruce.html">
    <img src="tartans/icons/bruce.png" alt="Bruce" />
    <h3>Bruce</h3>
  </a></li>
  <li><a href="tartans/buchanan.html">
    <img src="tartans/icons/buchanan.png" alt="Buchanan" />
    <h3>Buchanan</h3>
  </a></li>
</ul>
```

加入其余的花格布

还有更多的花格布需要增加到这个列表中（当然还包括它们漂亮的图标）。不用担心，我们不会让你输入太多代码。可以在extras文件夹中找到一个tartan-list.txt文件。在这个文件中，你会发现一段HTML代码，这是一个包含所有花格布的完整的<ul>。把这个<ul>复制粘贴到tartans.html中，替换掉当前的HTML列表。

现在我们有了一个增强的列表（嘿，有漂亮的图标），而且包括当前收藏的所有花格布。



嘿，列表看上去很不错，不过有点长。如果不使劲向下滚动，很难找到一个特定的花格布名字。



确实是这样。这有点费劲。

事实上，jQuery Mobile还能帮我们做更多事情。对于我们的花格布列表，更易于实现、更有意义的改进就要到来了！

过滤和组织列表

1 **可以增加列表分割条。**

通过使用列表分割条（divider），可以把列表分割成多段。这有助于组织列表，将花格布按其首字母分组。

2 **可以增加一个列表过滤器。**

利用jQuery Mobile，可以很容易地为列表引入过滤器。过滤器看起来就像是一个搜索域（有一个小放大镜之类的图标），用户在这个域中输入时，将对列表进行过滤。这里完全不需要编写任何JavaScript代码！

要为列表增加过滤器，需要为<ul>增加data-filter属性，属性值为"true"。

对，就这么简单！

1

```
<ul data-role="listview" data-filter="true">
```

可以增加data-role属性为"list-divider"的<li>元素，从视觉上分割列表。

2

```
<ul data-role="listview" data-filter="true">
  <li data-role="list-divider">A</li>
  <li><a href="tartans/abercrombie.html">
  <img src="tartans/icons/abercrombie.png" alt="Abercrombie" />
  <h3>Abercrombie</h3>
</a></li>
```

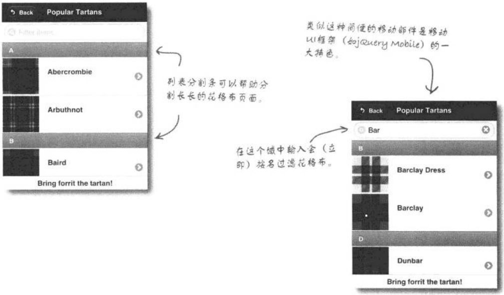
tartans.html

试一试

为tartans.html中的<ul>指定一个过滤器域和一些列表分割条。为每个字母增加一个列表分割条（除非没有以该字母开头的花格布，如Q和X）。

保存所做的修改，在浏览器中查看结果。很棒，是不是？

现在我们的花格布列表更棒了



there are no Dumb Questions

问：我注意到，你改变一个jQuery Mobile Web应用中的页面时，会有一种动画效果。那是怎么回事？

答：为了让Web应用感觉与移动用户界面模式更一致，jQuery Mobile对页面变换应用了一种过渡效果。

默认会使用幻灯片过渡，新页面将从右边滑入。要改变所使用的这种过渡，可以对当前链接增加一个data-transition属性。它支持大约6种不同的过渡效果。有关的详细信息参见jQuery Mobile文档（见其网站）。

问：这个特性在所有手机上都能用吗？

答：没有哪种移动框架能做这种保证。不过jQuery Mobile在努力尽可能提供更多的跨平台支持。可以在http://jquerymobile.com/gbs看到所支持的设备和浏览器列表，了解它们目前的支持程度。

问：如果一个设备或浏览器没有得到支持会怎么样呢？或者，如果关闭了JavaScript呢？

答：jQuery Mobile非常偏重渐进增强。我们的底层标记就采用了这种技术！

你何不自己试试看呢？在一个禁用了JavaScript的Web浏览器中加载Tartanator。当然了，它并不漂亮，不过确实能正常工作。

向Ewan展示Tartanator的早期阶段成果

我们的项目完成几个步骤了？

- [x] 构建内容页面和网站结构。
- [x] 创建花格布列表。
- [ ] 构建花格布创作表单的一个原型。

开始创建表单之前，先与客户沟通一下，确保目前为止整个方法没问题。





Jim："让它感觉更像一个原生应用"，这到底是什么意思？

Frank：感觉这是一个非常主观的要求，不是吗？我想Ewan寻找的是一种感觉更像"应用"的东西。

Jim：这是指……？

Frank：我想是不是像标签页、导航元素、按钮和图标之类的东西。比如，看来Ewan对这种过渡效果很满意。对他来说，这些感觉就是原生的。

Jim：不过我们不是在冒险吗？这样会不会过多地效仿某一个平台而忽略了其他平台呢？

Frank：嗯，没错。人们谈到"原生"时，有时他们只是想表示"让它看起来像iOS"，这可能很危险。每个平台都有自己的UI标志。

Jim：听起来就像"第22条军规"。为了让顾客满意，我们要让它看起来像原生应用，而这可能会怠慢另外一些用户，比如Android和黑莓手机的用户。

Frank：我想我们需要对某些方面进行改善，让Tartanator感觉更像是一个应用，而不是让它看起来像是一个iPhone应用。坦率地讲，jQuery Mobile元素本身看起来确实有点iOS范儿。所以我们已经有一定的基础了。

Jim：这么说，现在的目标是让Tartanator感觉更像是一个应用，而没有必要让它看起来是特定于某个平台的原生应用，是吗？

Frank：完全正确。我会把我的想法先写下来，我想可以做几个修改来达到这个目标。

新目标：让Tartanator看起来更像是一个应用，而没有必要看起来特定于某个平台。

让它更像应用的技巧

教你几招，我觉得完成这几步之后，可以让Tartanator感觉更像是一个应用。





更像是一个应用！

—把当前的链接<ul>替换为总在屏幕最下方的一个toolbar/navbar/（导航栏）。

—工具栏上带图标的按钮会让人感觉更像一个应用。

—页眉也总出现在固定位置。

"关于我们"页面呢？

—Ewan最近发邮件来说：这个页面的内容还没有收集到，让他很失望。

—是不是该把登录页面和"关于我们"页面合并为一个页面？把登录页面用作信息页面怎么样？

让Tartanator感觉更像应用的任务列表

- [ ] 把登录页面上的链接转换为带图标的工具栏，总在固定位置出现。
- [ ] 去掉单独的"关于我们"页面，因为这个页面的内容还没有得到。现在考虑把当前的登录页面作为"关于"页面。
- [ ] 让页眉也一直出现在固定位置。

增加一个页脚工具栏

在jQuery Mobile中，很容易在页眉或页脚中增加一个固定位置的工具栏。为了更像一个应用，下面在页脚放置一个工具栏，而不是在登录页面的<ul>中给出指向Tartanator各板块的链接。

构造导航栏

在页眉或页脚容器中（由data-role属性指示），我们可以再放一个data-role为navbar的<div>。这就告诉jQuery Mobile要把这些内容处理为工具栏中的按钮。基本构造如下所示：

注意我们删除了data-theme属性。

data-role为navbar，这会触发jQuery Mobile使它看起来像是一个（嗯，顾名思义）导航栏。

导航栏按钮（即链接）将放在这里。

```
<div data-role="footer" data-position="fixed">
  <div data-role="navbar">

  </div><!-- /navbar -->
</div><!-- /footer -->
```

index.html

在导航栏中放入按钮

我们不再用一个垂直列表组织链接，现在要通过工具栏按钮访问Tartanator的主要板块。这种方式会更像一个应用。

jQuery Mobile会自动将导航栏中的列表元素转换为按钮。

要记住，我们去掉了aboutus.html页面，现在使用index.html页面作为About页面。

jQuery Mobile会自动使这些链接看起来像按钮。

```
<div data-role="footer" data-position="fixed">
  <div data-role="navbar">
    <ul>
    <li><a href="index.html">About</a></li>
    <li><a href="findevent.html">Events</a></li>
    <li><a href="tartans.html">Tartans</a></li>
    </ul>
  </div><!-- /navbar -->
</div><!-- /footer -->
```

index.html

让工具栏更时尚

到目前为止还不错，不过我们的工具栏按钮有点单调。jQuery Mobile提供了一组大约18个默认图标，可供使用。下面使用data-icon属性放入一些图标。

目前的按钮显示效果。



它们看起来像按钮，还不错，不过还可以更洋气一点。

可用的默认图标列表可以在jQuery Mobile文档中找到。

data-icon属性指示这个按钮要使用哪个图标。

```
<div data-role="footer" data-position="fixed">
  <div data-role="navbar">
    <ul>
    <li><a href="index.html" data-icon="info">About</a></li>
    <li><a href="findevent.html" data-icon="star">Events</a></li>
    <li><a href="tartans.html" data-icon="grid">Tartans</a></li>
    </ul>
  </div><!-- /navbar -->
</div><!-- /footer -->
```

index.html

另外，下面在适当的链接上设置一个ui-btn-active类，指示用户当前正在查看哪个版块。按照主题设置会将它突出显示，这样看起来这就是当前的活动链接。

这个代码放在index.html中。在其他页面中，可能需要把这个类指定到适当的锚标记。

```
<ul>
<li><a href="index.html" data-icon="info" class="ui-btn-active">About</a></li>
<li><a href="findevent.html" data-icon="star">Events</a></li>
<li><a href="tartans.html" data-icon="grid">Tartans</a></li>
</ul>
```

index.html

最终确定结构



更新代码后显示的导航栏。

再来查看我们的任务列表：

- [x] 把登录页面上的链接转换为带图标的工具栏，总在固定位置出现。
- [x] 去掉单独的“关于我们”页面，因为这个页面的内容还没有得到。现在考虑把当前的登录页面作为“关于”页面。
- [ ] 让页眉也一直出现在固定位置。

现在只需要稍稍调整页眉……

我们花了那么长时间调整的页脚口号呢？



噢，对，Web上就是这样！需求总在变。

我们已经把页脚换成了一个导航栏，不过不用把这个口号完全丢掉。下面让它作为登录页面上的一个子标题（只在登录页面上出现）。

让页眉也出现在固定位置

既然已经做到这一步，下面让页眉也有固定的位置。类似于页脚，即使用户滚动界面，页眉也总是出现在屏幕最上面。

```
<div data-role="header" data-position="fixed">
  <h1>The Tartanator</h1>
</div><!-- /header -->
<div data-role="header" data-theme="b" class="forrit">Bring forrit
the tartan!</div>
```

index.html

试一试

哈，交互式设计。这正是Web之道。下面更新Tartanator。

1. **实现新的页脚导航栏。**

   编辑index.html、findevent.html和tartans.html。把页脚替换为我们在250页上完成的版本。

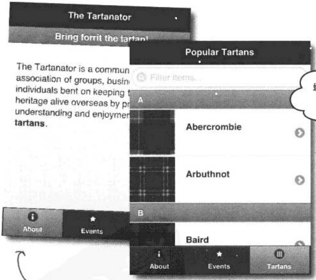
   根据具体的页面，确保为正确的锚标记指定ui-btn-active类。

2. **为页眉指定一个固定位置。**

   为每个页面上的页眉增加一个data-position属性。

3. **为登录页面（index.html）增加口号。**

   为index.html增加口号作为一个子标题（只在这个页面上增加）。



页眉和页脚现在都是固定的。它们总是（分别）出现在页面的最上面和最下面，即使用户上下滚动，它们的位置也不变。



建立花格布创作表单

对于这个项目，下一步是创建一个表单原型，允许用户在他们的移动浏览器上设计自己的花格布。

- [x] 构建内容页面和网站结构。
- [x] 创建花格布列表。
- [ ] 构建花格布创作表单的一个原型。

  Ewan希望用户能够使用一个类似应用的移动界面，创作他们自己的花格布。



花格布设计就像是一个菜谱。

不过，并不是一个食物配料及其用量（盎司、克、匙等）的列表，这个列表会列出不同颜色及各种颜色在图案中的相对大小。织花格布时，会在水平和垂直方向重复编织这个图案。

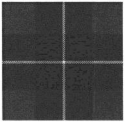
我们要大致了解花格布是如何合成的，这样才能建立一个表单，收集适当的信息。

花格布：图案就像菜谱

下面举个例子。要创建Carmichael clan花格布，图案如下所示：

图案中的各个成分是一个“颜色-大小”对。

颜色	大小（针）
黑	6
绿	72
蓝	56
红	4
蓝	4



Carmichael clan花格布。

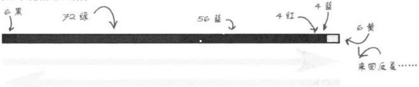
花格布配方

要织这块花格布，需要按顺序依照图案编织（这里我们要用像素而不是毛线来“编织”）。

对于Carmichael，先织6针黑色，然后是72针绿色，接下来是56针蓝色，如此继续。

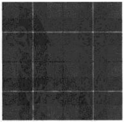
达到图案中的最后一个颜色时（在这个例子中就是6针黄色），再按逆序编织图案（蓝，红，蓝，绿）。再次达到第一个颜色(黑色)时，再循环反复。水平和垂直方向（经线和纬线）都按这个图案编织，最后得到整个花格布图案。

第一个和最后一个颜色（黑和黄）不会重复。对于Tartanator，这些被称为图案中的“轴心”。



嘿……这就是上面的图案，只不过这里是水平而不是垂直显示。

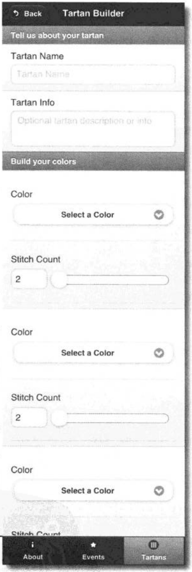
不同颜色在水平和垂直方向上重叠，会创造亮丽的图案。



我们要创建表单域，收集用户花格布设计中各个成分的颜色和大小值。

把花格布图案转换成一个表单

下面来看我们想构建的这个表单原型。我们需要一些域，允许用户输入颜色-大小对，来建立一个花格布配方。



所创建的花格布的基本信息：花格布名字，还可以有一个描述（可选）。

颜色-大小域对，利用颜色-大小组合域……

……用户可以建立其花格布配方的成分列表。

还可以有更多。

OK。我们已经基本明确了要构建的表单。下面就来构建这个表单！

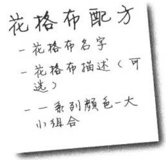
Exercise

创建一个jQuery Mobile空页面，在其中包含这个花格布创作表单。参考tartans.html创建一个空页面。使用同样的页眉和页脚，不过把页面的标题和页眉改为“Tartan Builder”。把这个文件命名为build.php。

构建一个HTML5表单

表单本身是一个标准的HTML5表单：jQuery Mobile会对它进行调整，使它感觉是移动友好的。在第7章中，我们将让这个表单完成具体工作，包括利用JavaScript增强其交互性，并生成真正的花格布图像。

对现在来说，我们只打基础，创建一个没有花哨装饰的基本表单，让它在几乎所有移动浏览器中都能正常工作。



这些就是我们的表单要收集的信息。

表单结构

我们希望用户能够对他们的花格布命名，另外还可以输入一个描述（可选）。然后，可以定义颜色-大小组合来建立图案本身。下面把表单分为两个主要部分：上半部分收集关于花格布的元数据，另外一部分用于定义图案的颜色和大小。

成品表单结构

我们将使用一个listview 帮助建立表单布局。

通过使用ID属性，以后我们可以编写脚本增强这个表单。

```
<div data-role="content">
  <form id="tartanator_form">
    <ul data-role="listview" id="tartanator_form_list">
      <li data-role="list-divider">Tell us about your tartan</li>
      <li data-role="list-divider">Build your colors</li>
    </ul>
  </form>
</div><!-- /content -->
```

build.php

这两个list-divider 把表单分割成两部分。

增加一些基本的域

元数据表单域用来输入花格布名字和描述，这些表单域很基本。我们将分别使用一个文本input和一个textarea域实现。这里使用的标记和你以往常见的几乎没有什么差别，只不过这里利用了HTML5的placeholder属性。

另外经过精心设计，我们还在适当的地方使用了可访问的label元素。

利用placeholder属性，我们可以在输入域中增加初始的占位文本；用户点击这个域并修改其内容时，占位文本将会“消失”。



成品表单域

```
<li data-role="list-divider">Tell us about your tartan</li>
  <li data-role="fieldcontain">
    <label for="tartan_name">Tartan Name</label>
    <input type="text" name="name" id="tartan_name"
placeholder="Tartan Name" />
  </li>
  <li data-role="fieldcontain">
    <label for="tartan_info">Tartan Info</label>
    <textarea cols="40" rows="8" name="tartan_info"
id="tartan_info" placeholder="Optional tartan description
or info"></textarea>
  </li>
  <li data-role="list-divider">Build your colors</li>
```

build.php

为jQuery Mobile提供域的提示信息

你可能已经注意到，上面各个域都分别在一个data-role为fieldcontain的元素中。data-role为fieldcontain就是向jQuery Mobile提供一个提示，指示这个元素中的表单域可以进一步增强。

我们使用了一个<ul>组织表单，所以这里为包含输入域的各个<li>指定fieldcontain类。

嵌套列表允许用户增加颜色

花格布图案的每一部分都是一个颜色和一个大小（宽度）的组合。建立布局时，我们将把每一对颜色和大小组合域归组到一个<li>中。为了保证提供最基本的用户体验（也就是说，在不支持JavaScript的浏览器上也能正常工作），我们将用PHP生成6个这样的表单域组（这样可以少输入代码）！

下面的代码可以作为起点：

所以，每一对颜色-大小域分别包含在一个<li>中，也就是说，每个<li>有两个表单域。

```
<li data-role="list-divider">Build your colors</li>
  <?php for ($i = 0; $i < 6; $i++): // 6 color fields ?>
  <li class="colorset">

  </li>
  <?php endfor; ?>
</ul>
```

这个PHP for循环会循环6次，生成6个表单域组。

颜色和大小域放在这里。

build.php

there are no Dumb Questions

问： 为什么没有把这个<li>的data-role指定为fieldcontain类？它里面也有域啊。

答： 每个这样的<li>（li.colorset）实际上包含两个域。构建这些域本身时，会把它们分别放在其自己的<div>中，并把那个<div>的 data-role指定为fieldcontain。

问： 我糊涂了。该为哪种元素指定fieldcontain data-role属性呢？

答： 只要元素中包含域本身，也就是说，域的直接父元素都可以指定这个属性。比如<li>或<div>或<fieldset>或<p>（或者诸如此类的元素）。

如果data-role为fieldcontain，就是在告诉jQuery Mobile：增强表单时要把这个元素包含的域及其标签组合起来。因此，每个data-role为fieldcontain的包含元素都应该包含一个域（及其label）。

问： 需要使用PHP来完成项目的这一部分吗？

答： 没错！我们就要进入Tartanator的功能实现部分了，接下来将使用PHP来完成设计和处理。

颜色-大小域对：颜色选择域

每组颜色域都有一个颜色（<select>）和一个大小输入域。这里指定了类color-input和size-input，以后用JavaScript增强表单时会用到这些类。

如果能使用HTML颜色输入域就好了，不过很可惜，这在很多浏览器中都未得到支持。

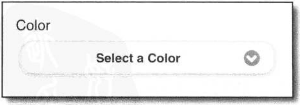
显然，我们不希望只有黑色和白色可以选择！有关的更多内容稍后讨论。

```
<li class="colorset">
  <div data-role="fieldcontain" class="color-input">
  <label class="select" for="color-<?php print $i ?>">
  Color</label>
    <select name="colors[]" id="color-<?php print $i ?>" >
      <option value="">Select a Color</option>
      <option value="#000000">黑</option>
      <option value="#ffffff">White</option>
    </select>
  </div>
</li>
```

应该记得，这是在一个PHP循环中，这会为各个域赋一个唯一的ID。

build.php

在基于WebKit的现代移动浏览器上，上面的域会如此显示。



颜色-大小域对：大小域

我们使用了一个HTML5新增的输入类型：range。这个域类型在现代浏览器中得到了越来越广泛的支持（不过当然还没有完全普及）。在支持这个类型的浏览器中，它会显示为一个滑动条；在不支持的浏览器中，会降级显示为一个文本域。jQuery Mobile进一步增强了这个滑动条，让它变得更加移动友好。

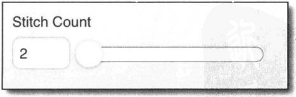
这部分就放在我们刚才构建的<select>后面。

min和max属性的作用顾名思义，定义域的最小和最大值。

```
    </select>
  </div>
  <div data-role="fieldcontain" class="size-input">
    <label for="size-<?php print $i ?>">Stitch Count</label>
    <input id="size-<?php print $i ?>" type="range" min="2"
step="2" max="72" autocomplete="off" name="sizes[]" value="2" />
  </div>
</li>
```

花格布图案的针数只能是2的倍数。对于这一点，step属性应该对我们有帮助。

build.php



在基于WebKit的现代移动浏览器上，上面的域会如此显示。

试一试

把它们集合起来，构建一个有移动外观（尽管还没有实际功能）的表单。

1. **在build.php中加入表单域。**

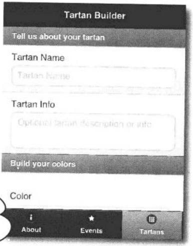
   加入256页到260页的标记，创建基本的表单域。

2. **为颜色域增加更多颜色选择。**

   可以使用extras/color-list.txt文件中的颜色（十六进制值），并（或者）增加你喜欢的任何颜色来完成花格布创作。

保存build.php，在一个移动浏览器中加载！

外观（感觉）更像一个应用。





我们需要链接到表单，然后再以一种类似应用的方式返回。现在就开始吧！

链接到表单

这个创作表单是Tartanator中花格布板块的一部分。我们要为用户提供一条路，能够从花格布登录页面（有一个长长的列表，列出了所有现有的花格布）进入这个表单。在页眉增加一个按钮应该就足够了：

```
<div data-role="header" data-position="fixed">
  <h1>Popular Tartans</h1>
  <a href="build.php" data-role="button" data-icon="plus"
class="ui-btn-right" data-theme="b">Create</a>
</div><!-- /header -->
```

tartans.html

为按钮指定一个“加号”图标。

ui-btn-right类使按钮浮动到右边（而不是默认的左边）。

指定主题swatch为“b”，这会让按钮更突出一些。

……再为用户提供一条回来的路

再在表单页面（build.php）上增加一个返回按钮：

```
<div data-role="header" data-position="fixed">
  <a href="tartans.html" data-rel="back" data-icon="back"
data-role="button">Back</a>
  <h1>Tartan Builder</h1>
</div><!-- /header -->
```

build.php

通过为锚标记增加data-rel="back"，这会告诉jQuery Mobile要把这个链接当作一个返回链接。如果可能，用户单击这个链接时，jQuery Mobile会把用户发回到其历史列表中的上一个位置。另外对于不支持这个特性的浏览器，我们提供了一个很好的默认链接地址，在这里就是tartans.html。

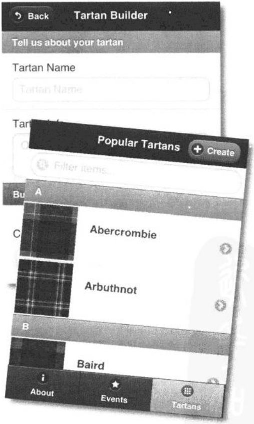
试一试

1. **从tartans.html链接到表单。**

   在tartans.html中为页眉增加一个“创建”（Create）按钮。

2. **在表单页面(build.php)上增加一个返回按钮。**

   在Tartan Builder页面的页眉上增加一个返回链接。



☑ 构建内容页面和网站结构。

需要为网站创建基本页面，并创建一个在移动设备上感觉很好的整体结构。

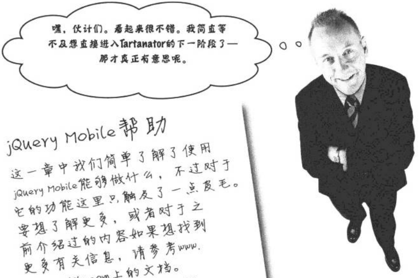
☑ 创建花格布列表。

对于第一阶段，我们将创建一个包含现有流行花格布图案的列表。花格布板块应当是一个浏览界面，当然，感觉要像个应用，而且要面向移动设备。

☑ 构建花格布创作表单的一个原型。

最后，Ewan希望用户能够使用一个类似应用的移动界面创作他们自己的花格布。他想看看这个界面是什么样子，所以我们要为他设计一个原型。

耶！我们已经完成了Tartanator的第1阶段。现在可以让客户检查我们的成果了。



jQuery Mobile帮助

这一章中我们简单了解了使用jQuery Mobile能够做什么，不过对于它的功能这里只触及了一点皮毛。要想了解更多，或者对于之前介绍过的内容如果想找到更多有关信息，请参考www.jquerymobile.com上的文档。它提供了非常详细的介绍。

BULLET POINTS

- 人们想得到真正的应用！怎样的网站才能算是一个Web应用呢？这方面的定义很含糊。与内容页面相比，类似应用的网站往往更有交互性。内容和数据块通常会异步获取，然后插入到现有的DOM中，从而尽可能减少整个页面的加载。
- HTML5很明确一它是一个规范，表示了超文本标记语言（HyperText Markup Language，HTML）的演进。不过HTML5这个词通常还用来表示创建应用型web体验的一个技术组合。
- 从头开始构建移动Web应用可能非常复杂。建议你自己动手试试看！不过，对我们来说，这里使用了一个移动用户界面开发框架来助我们一臂之力。
- 现在有很多移动Web框架（每天还有更多框架产生），分别表示不同的方法，强调不同的关注点。
- jQuery Mobile是一个流行的移动Web框架。它与良构的HTML5标记关系很紧密，这使得由基本代码构建移动界面相当简单。
- 在Tartanator的第1阶段，我们使用jQuery Mobile（以及其他技术）构建了网站结构，包括增强的listview、页眉、页脚、导航栏和表单元素。
- 要让一个web应用感觉更像是原生应用，这往往是一种讲究平衡的艺术，要求精心考量。
- 构建表单原型时我们使用了一些HTML5表单元素属性。jQuery Mobile会在不同移动浏览器上调整这些表单元素，包括为那些不提供支持的浏览器做适应性调整。

there are no Dumb Questions

问：到目前为止，Tartanator中有多少代码是真正的HTML5？也就是说，其中哪些使用了HTML5规范新增的标记或属性？

答： 我们使用了很多data-*属性。这些就是新增的。我们还在表单中使用了range输入类型和placeholder属性。

问：对于表单中的大小域，我为它的range输入域增加了step属性，不过仍然能输入2到72之间的任何值……奇数和偶数都能输入。这是怎么回事？

答： 很遗憾，写这本书时，jQuery Mobile的range滑动条部件还不支持step属性。我们会在第7章提供一个解决方案。

问：这里只是通过列表来选择颜色，这可不算是一种有冲击力的用户体验。

答： 第7章中将增强这个表单，我们会增加一些JavaScript代码，向用户展示他选择的颜色。

问：不支持range输入域的浏览器会怎么样呢？

答： 只要浏览器能很好地支持JavaScript和CSS，jQuery Mobile就能把range域转换为一个滑动条部件（实际上，对于支持range输入域的浏览器也会这样做）。

如果一个浏览器不支持range输入域，而且没有JavaScript支持（或者禁用了JavaScript支持），很可能会把range域显示为一个标准的文本输入域。

问：页眉和页脚有时候会闪烁，或者比页面其他部分加载得慢。有时滚动感觉有点慢，为什么会这样？

答： jQuery Mobile模仿原生体验的一种方法是使用固定位置的页眉和页脚。在不同的平台上，具体实现的方法各有不同。

需要说明的是：尽管这个领域发展得非常快，但移动浏览器中对固定定位（在CSS中，position:fixed）的支持却不尽人意，很分散，甚至很奇怪。对此提出了一些解决办法，不过并不完美。因此，应用性能偶尔会下降，另外有时会有特异的表现，这些通常就是框架权衡折中的体现，一方面力图更像应用，另一方面又要坚持Web标准，于是就可能出现你提到的这些情况。



祝贺你，你已经用HTML5和它的朋友们构建了一个移动Web应用。

看到了吧？在你掌握的HTML和CSS知识基础上，建立一个类似应用的移动网站并不需要太多复杂的步骤！

7 真实世界的移动Web应用

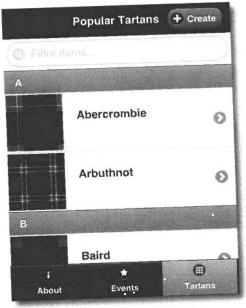
超级移动Web应用



移动Web就像班里那个小机灵鬼。你知道的，就是那种很让人喜欢、聪明能干的小家伙，不过也是捉摸不定、让人防不胜防的淘气包。我们尽量不设置太多约束和条条框框，希望不扼杀他们过人的天赋，不过现在该是有效利用移动Web天分的时候了。利用渐进增强，我们可以在更成熟的浏览器中把原本简朴的界面装扮一新。通过建立一种周全的离线模式，能够让飘忽不定的网络连接从一种负担转变成一个抢眼的特性。另外，通过使用地理定位可以得到移动的真谛。下面就来建立这样一个超级移动Web应用吧！

看着还不错……

现在我们已经完成了第1阶段，Tartanator已经很有一副移动Web应用的派头。



嘴上说得好……

这里有按钮。还有导航栏。另外有很酷的页面过渡。我们用Ajax有选择地加载内容，减少了带宽、请求和JavaScript DOM处理。另外在jQuery Mobile的帮助下，我们的HTML5表单元素也感觉是移动友好的。



别灰心。你可能觉得第1阶段只是加了一些花哨的装饰，没有实际的东西。不过我们确实打下了坚实的基础。

我们已经有一个基于HTML5的简单结构，现在可以让它更上一层楼！

……不如做得好

第2阶段会在之前工作的基础上，把这个强健的Web应用（不过不能不承认，它还没有具体的功能）变成我们设想的一个超级移动应用！

真实世界的移动应用

能充分利用固有移动特性的移动Web应用通常有某些共同点。

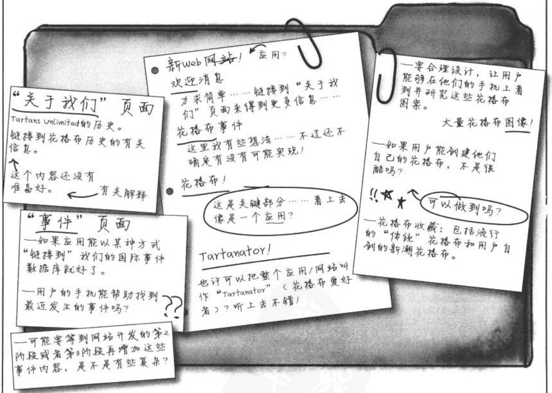
我们所称的超级移动Web应用就像是真实世界的移动应用。通过稳妥地使用渐进增强，它们能适应用户不同的设备。用户没有数据连接时，这些应用可以采用一种离线模式工作。而且它们充分利用了浏览器支持的地理定位来提供与位置相关的内容。



Sharpen your pencil

怎么完成第2阶段，使Tartanator成为一个超级移动Web应用？

我们的第一个任务是搞清楚需要做什么才能实现Ewan在第6章描绘的Tartanator。重点关注我们还没有完成的需求，然后把它们汇集起来，作为这个阶段的核心目标。



Sharpen your pencil Solution

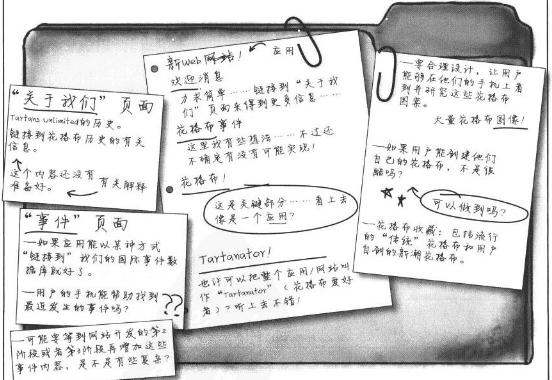
这个项目第2阶段的主要目标实际上可以归为两大核心目标：

1. **加入代码，使用户能生成他们自己的花格布。**

   我们已经建立了原型表单。现在需要落到实处，加入代码，让这个表单能真正起作用。我们还需要增强现有表单，让它在更时尚的智能手机上更有用，另外增加一些小部件。

2. **构建一个动态、可搜索的与位置相关的事件页面。**

   我们需要关联已有的一个事件数据源，允许用户查找附近与花格布有关的事件。这说明我们要用到地理定位！





Frank：第一个大需求混合了很多小需求，不是吗？我想把它分成更小的代码块会有帮助。

Jim：先别管这个，现在想想“关于我们”页面和历史背景的那些内容怎么办？

Frank：噢，对了！为Tartans Unlimited整理自传和信息的那个人刚做了爸爸。他发现比原来预想的要忙多了，他暂时分身乏术，没办法来帮忙。也算是个惊喜，是吧？所以我们现在先不用考虑这个问题。

Jim：嗯，我想这样可以少一件事。下面来讨论怎么加入代码，允许用户创建他们自己的花格布图案。不是在吹牛，我在这方面已经做了一些工作。我一直在写JavaScript来改进表单。

Joe：确实很酷。看起来很漂亮。不过我们还需要加入服务器端脚本来做具体的工作。

Jim：你们这些人呐，张口闭口都是服务器端！相信我，增强表单界面绝对是个显著的改进。

Joe：不过要知道，我一直在编写PHP脚本。那些事件呢？

Jim：实际上，我们是不是可以先重点考虑花格布生成，等以后感觉做好准备了再考虑事件搜索，那时再回过头来研究这个问题，怎么样？我头脑有限，一次只能考虑一个问题。

Frank：我想，只要能快点让用户创作花格布就行。我们的时间很紧。

任务列表：定制用户花格布图案实现

这是实现定制花格布需求的任务列表。

我们从这里开始！

- [ ] 增强我们之前构建的表单，充分利用新型移动浏览器的功能。
- [ ] 加入服务器端代码来处理表单，并生成用户自创花格布的相关资源（图像、HTML等）。
- [ ] 确保应用这一部分的离线体验能让人接受。

各就各位，预备，增强！

做好准备！稍后我们就要对Tartanator的创作表单做很多漂亮的增强处理。

我们已有的基础

我们已经设计了一个表单，可以提供最基本的用户体验：我们不会冷落任何人（嗯，也许极少数人除外）。你可以在一个根本不支持JavaScript的浏览器上加载Tartanator创作表单，看看会发生什么。

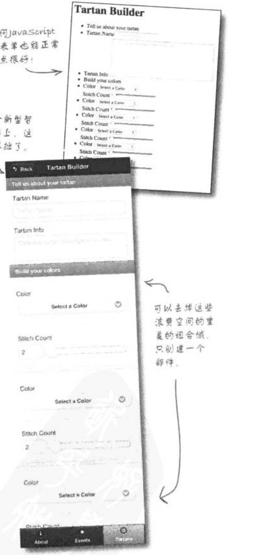
现在来增强这个表单

这个表单对所有人都适用，不过另一方面讲，在所有人看来它都很丑陋，也很怪异。对于一个功能不太全面的浏览器，有6个颜色-大小组合域尚能接受，但是对于功能更强的浏览器来说，这就很是累赘了。

既然我们已经满足了基本需求，下面来做一些增强处理，让表单在智能手机上使用时成为一种享受。

即使没有任何JavaScript支持，这个表单也能正常工作，这一点很好！

不过，在一个新型智能手机浏览器上，这就显得有些笨拙了。



可以去掉这些浪费空间的重复的组合域，只创建一个部件。

实现超级移动Web应用的第一步：对能够提供支持的浏览器完成UI增强。



不要一听说编写你自己的部件就惊惶失措。我们已经为你提供了相应的JavaScript。

一些一流的高级开发人员已经为你写好了这个增强JavaScript。对于移动Web应用项目，他们总是热情高涨，工作完成得又快又好！

我们会带你了解其中的重点内容（你也可以花一些时间研究这些代码），不过你不需要自己动手写任何代码！

建立一个更棒的表单

这6个域挤在一起很不美观，而且功能有限，所以现在不再用这6个域，我们只使用一个部件。这样一来，用户可以随心所欲地增加更多的颜色-大小组合域！为此，除了第一组外，可以使用JavaScript删除所有其他的颜色-大小域。

定制颜色选择域部件

我们可以覆盖浏览器中的默认选择界面，使用jQuery Mobile的一个定制选择UI部件。这样一来，还可以为每个选择显示颜色块。

我们要用的定制jQuery Mobile部件将像对话框一样弹出，显示颜色选择。

只有一组颜色-大小域，而不是6个。



单击"Select a Color"（选择一个颜色）时，会弹出这个定制域部件。

如果想了解jQuery和JavaScript，可以看看下面的解释。



Geek Bits

如果你熟悉jQuery代码，可能已经多次见过（或用过）$(document).ready构造。jQuery .ready方法就类似于浏览器的body.onload事件（但不完全相同），触发这个事件时，会指示DOM已经加载并准备就绪。jQuery开发人员通常会把代码包围在这个事件中，保证在页面DOM准备好之前不会执行这些代码。

在jQuery Mobile(jQM)中，（通常）不会使用$(document).ready。实际上，jQM引入了一些新的与页面加载相关的事件，其中最重要的是pageinit和pagecreate。jQuery和jQM完成页面DOM初始化之后但在显示部件之前会触发pagecreate。在部件完成之后则会触发pageinit。要记住，后续页面通常会插入第一个页面现有的DOM中，这说明在一个完整的页面加载中pagecreate和pageinit事件可以触发多次。

jQM还为触摸、方向和过渡等方面引入了与移动有关的事件。

是不是不明白？反正我们有点糊涂了。即使是资深的JavaScript和jQuery专家掌握这个概念也要花一些时间。有关的更多内容可以参考JQM文档（http://jquerymobile.com/demos）。

一个部件管理颜色大小列表

除了一个颜色-大小组合域外，我们已经删除了所有其他组合域。利用剩下的这个颜色大小域对，我们需要能够生成任意数量的颜色-大小组合域。

为此，我们增加了一个新按钮：“Add This Color”（增加这个颜色）。单击时，当前选择的颜色和针数会作为一个列表元素（<li>）增加到一个无序列表中（<ul>），这个无序列表包含当前的所有颜色-大小值。我们还要为<li>增加隐藏的表单域来包含颜色和大小。

所有这些工作由JavaScript完成。

单击现有颜色列表中的一个<li>，会将它及其内容从列表中删除。最后，单击“Make it!”提交按钮会生成一个花格布图案，其中包含列表中当前的所有颜色-大小。



增强的build.php表单（详细）

动手试一试

嘿，好消息。你不需要做任何事情就能得到这些前端增强效果。所有工作都已经帮你完成了。

可以把chapter7文件夹中的代码作为起点，其中已经包含这些JavaScript增强实现。可以试试看！

要记住，表单在build.php页面上。

另外要注意，我们还没有“关联”后端代码。单击这个提交按钮时什么也不会发生！

看看在一个移动浏览器中表现如何。

深度探秘

我们承诺过，不需要你写任何JavaScript代码，不过可以大致了解究竟是怎样做的，这可能对你有好处。

在底层，包含表单的页面完全加载到DOM时（pagecreate事件）会发生很多事情，另外在jQuery Mobile部件完成增强之后（pageinit事件）也会发生很多事情。

在这些初始化函数中，会把其他函数关联到不同事件（表单域改变、点击、表单提交等），来增强用户交互，使我们的定制部分真正起作用。

动手做！

熟悉这些代码。花几分钟仔细看看，了解Tartanator应用中哪些部分有更新，尤其是更新的tartanator.js脚本。

WHO DOES WHAT?

把JavaScript增强代码(tartanator.js)中的各个函数与相应的作用连线，另外还要连接触发这个函数的事件。可以参考脚本中的注释，同时利用你的直觉，确定各个函数究竟为我们做什么。

再加一个挑战！

函数名	作用	触发事件
onStitchSizeChange()	刷新当前颜色列表的 和 元素。	pageinit和change（颜色选择域）
styleColorListItem()	构造按钮来增加一个颜色-大小组合域，并将它插入到DOM中。	click（“Add This Color”按钮）
buildAddButton()	将选择部件的背景色设置为当前选择的颜色。	pageinit
onColorListChange()	构造隐藏的表单域，将它们放在一个新 中，并把这个 追加到现有的颜色列表中。	click（当前颜色列表中的列表元素），新颜色增加到列表之后
addColor()	在各个颜色选择项左边增加一个彩色CSS边框（swatch）。	change（大小输入值有更新）
setColorSelectStyle()	确保大小值是一个偶整数。	pagecreate

这就是前端增强……

- [x] 增强我们之前构建的表单，充分利用新型移动浏览器的功能。 ← 完成了！
- [ ] 加入服务器端代码来处理表单，并生成用户自创花格布的相关资源（图像、HTML等）。 ← 下一步！
- [ ] 确保应用这一部分的离线体验能让人接受。

WHO DOES WHAT? 答案

函数名	作用	触发事件
onStitchSizeChange()	刷新当前颜色列表的 和 元素。	pageinit和change（颜色选择域）
styleColorListItem()	构造按钮来增加一个颜色-大小组合域，并将它插入到DOM中。	click（“Add This Color”按钮）
buildAddButton()	将选择部件的背景色设置为当前选择的颜色。	pageinit
onColorListChange()	构造隐藏的表单域，将它们放在一个新 中，并把这个 追加到现有的颜色列表中。	click（当前颜色列表中的列表元素），新颜色增加到列表之后
addColor()	在各个颜色选择项左边增加一个彩色CSS边框（swatch）。	change（大小输入值有更新）
setColorSelectStyle()	确保大小值是一个偶整数。	pagecreate



界面增强闪亮登场

本周访谈：
增强：都只是装饰吗？

Head First：好吧，我得问问了。我们刚刚完成了一大堆jQuery Mobile特定的JavaScript界面增强。这真的有必要吗？

界面增强：简单的说，关于增强移动界面，jQuery Mobile的工作确实很抢眼，这是当然的，不过我懂你的意思。实际上，这里我要展示一个更宽泛的概念。

Head First：你要说什么？

增强：不论你是否使用一个框架，都可以先设计一个基准，然后在这个基准之上增强，这是一个很好的策略。

Head First：利用JavaScript吗？

增强：你可以想得更远一些……这比JavaScript更宽泛。我们要强调的是：先从基本开始，定义核心的体验或内容，然后再根据情况进行改进，让它更出色，也就是说，充分利用更强大的移动浏览器的功能。

Head First：通过增加那些小配件吗？

增强：我希望我的贡献不只是圆角和梯度。我们已经建立了一个功能有限的表单，而且比较丑陋，接下来可以为有条件的用户改进这个表单。

Head First：你已经覆盖了一个原生表单元素的显示。现在颜色选择域会使用一个定制的jQuery Mobile部件。是这种改进吗？

增强：改变原生表单控件确实有点遭人诟病。我不建议把这作为一般经验。不过，对于这个特定的情况，为了向用户提供视觉提示（颜色块），利用传统的选择域根本没办法提供我们需要的样式。

底层标记（<select>元素）在语义上仍然是合适的，不过我们要调整这个元素，让它更有用。当然，你说的也没错，关于原生表单控件的覆盖，我们确实应该仔细考虑。

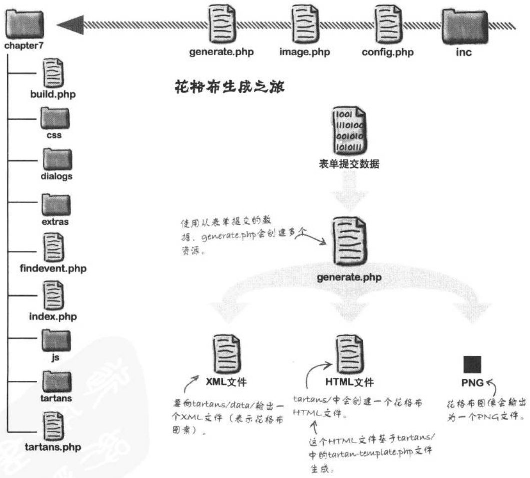


Android浏览器中的花格布创作表单。

……现在来看后端

第7章作为起点的代码有以下结构。要插入这些后端代码，只需要很简单的几步。

把extras/scripts目录的所有内容复制到你的chapter7目录。完成后，chapter7目录应该有3个新文件：config.php、generate.php和image.php，另外还有一个inc目录（其中包含两个文件）。



generate.php的两方面

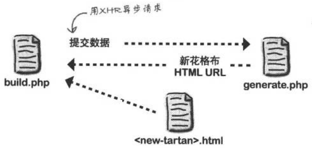
在创建花格布资源之后，generate.php最终如何响应，这取决于如何对它做出请求。

不论移动浏览器是否有Ajax支持，我们都要提供支持。

使用Ajax请求

对于能正确支持Ajax的浏览器，build.php中的JavaScript会使用XHR (XMLHttpRequest)把表单数据提交到generate.php。如果成功，generate.php会响应返回新创建的花格布HTML文件的URL。这个新页面的内容再插入到当前页面的DOM。

用XHR异步请求



在这个方法中，没有重新加载整个页面。新花格布页面的内容会插入到创作页面的DOM中。

这两种方法中，generate.php的主要工作是一样的：都要创建新花格布资源。

表单直接提交

对于不支持JavaScript和XHR的浏览器，表单会以一种“传统”方式直接提交到generate.php。generate.php创建花格布资源之后，浏览器会重定向到新构造的花格布页面。

“传统”方法——重定向到新花格布页面



在这个方法中，客户会重定向到新花格布页面（需要加载整个页面）。

最后一个问题！

既然用户可以增加花格布，tartans.php上显示的花格布列表就有可能一直在变。把下面这行代码放在tartans.php文件中，来包含一个脚本，它会为当前现有的各个花格布分别输出一个<li>。

动手做！

```
<ul data-role="listview" data-filter="true">
  <?php include('inc/list.php'); ?>
</ul>
```

tartans.php

Exercise

动手创建一些花格布！使用build.php上的表单创建你自己的5、6个花格布。

你的创作会显示在花格布登录页面上（tartans.php）。

there are no Dumb Questions

问： 快来帮忙！出问题了！

答： 如果花格布创作表单不能正常工作，或者没有看到你创作的花格布，可以检查以下几方面。

首先，确保tartans/目录存在，而且这个目录及其所有子目录对你的web服务器的用户都可写。另外，检查tartans目录中的tartan-template.php文件。最后，反复检查build.php中<form>标记的动作和方法是否正确。

问： 花格布列表页面是怎样工作的？

答： 这个页面中包含的list.php文件会查看tartans/目录中的当前HTML文件来生成列表。对于每个HTML文件，它会获取相关的XML文件（tartans/data/中）来得到更多信息，如漂亮的显示名。然后对每个花格布分别输出一个<li>，并输出一个链接指向这个花格布的HTML页面。

问： 相关的XML，这到底是什么？

答： 我们把花格布图案的数据表示存储为XML，这有几方面的原因。首先，这是一个简单的、可移植的格式。其次，这样你就不需要使用数据库了（可以少做很多工作）。

问： 顺便问一句，你说的这个超级移动Web应用到底是什么？标准？还是首创？或者是其他什么东西？

答： 不，我们只是这样称呼充分利用了移动设备及其浏览器强大功能的移动Web应用。

3个任务已经完成了两项，不错的开端

我们已经向超级移动web应用迈出了几大步。现在已经增强了基本界面，使它能利用更高级浏览器的强大功能。现在该让应用具体做点什么了！

- [x] 增强我们之前构建的表单，充分利用新型移动浏览器的功能。
- [x] 加入服务器端代码来处理表单，并生成用户自创花格布的相关资源（图像、HTML等）。
- [ ] 确保应用这一部分的离线体验能让人接受。

接下来需要处理这个需求。

不过还没有彻底完成！

现在必须完成花格布创作实现的第3部分：这些花格布要离线可用。



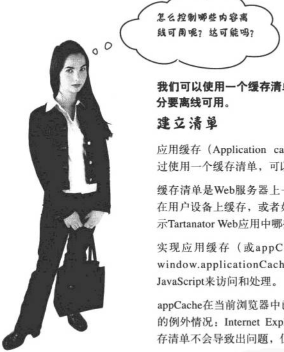
对于移动设备，无法保证有一个稳定的网络连接（甚至不能保证网络连接的存在）。

我们要采取一些措施，使得没有互联网连接时花格布也可用。

离线很重要

要提供让人满意的用户体验，其中很重要的一方面就是要让你的网站和应用在没有互联网连接时也有很好的表现。

对于Tartanator，我们需要后退一步，确定对于没有连接的用户如何更好地完成工作。我们要确保某些内容离线可用。



我们可以使用一个缓存清单（cache manifest）来定义应用的哪些部分要离线可用。

建立清单

应用缓存（Application cache）是HTML5规范中引入的一种缓存机制。通过使用一个缓存清单，可以控制哪些Web资源要缓存，从而能离线可用。

缓存清单是Web服务器上一种特殊类型的文件，指示某些web资源是否要在用户设备上缓存，或者如何缓存。通过创建一个缓存清单，我们可以指示Tartanator Web应用中哪些方面要离线可用。

实现应用缓存（或appCache，通常都这样简写）的浏览器还提供了window.applicationCache对象及其各个事件，我们可以根据需要利用JavaScript来访问和处理。

appCache在当前浏览器中已经得到相当广泛的支持，不过有几个比较突出的例外情况：Internet Explorer 9和Opera Mini。对于这些浏览器，使用缓存清单不会导致出问题，但这些浏览器不会注意到缓存。

BRAIN POWER

很难让Opera Mini支持应用缓存，你能想到这是为什么吗？

创建缓存清单的基本步骤

要在一个网站或应用上创建和使用缓存清单，一般有3个步骤。

1. 编写一个缓存清单文件。
2. 为支持缓存的页面的<html>标记增加一个manifest属性，并指定一个URL（也可以是相对URL），指向之前创建的清单文件。
3. 确保提供的清单有正确的文件类型。 ← 这个文件的content-type必须是text/cache-manifest，否则不能正常工作。

缓存清单的简单语法，不过这是假象

一个简单的缓存清单文件外表看起来相当基本。可以把想要缓存的项列在CACHE:标题下面（理论上不需要，不过这确实是一个很好的实践做法）。

这一行要原样保留，这是必需的。

1

```
CACHE MANIFEST
# comments look like this
CACHE:
index.html
foo/bar.html
baz.html
css/styles.css
icons/plus.png
```

URL可以是相对的（如这里的相对URL），也可以是绝对的（如http://…）。

manifest.appcache

这个文件没有必要非得命名为manifest，不过扩展名必须是.appcache。

接下来，更新Web页面，增加一个manifest属性指向这个清单文件。还要确保文件有正确的 Content-type（即MIME类型）。对于Apache，通常只需要向Apache配置文件增加这行代码，或者更常用的方法是向网站的文件系统增加一个.htaccess文件。

2

```
<!DOCTYPE html>
<html manifest="manifest.appcache">
<head>
```

3

```
AddType text/cache-manifest .appcache
```

.htaccess

这行代码告诉Web服务器，要提供扩展名为.appcache的文件作为text/cache-manifest内容类型。

这个文件必须是这种类型，否则浏览器无法识别。

遗憾的是，细节会出问题

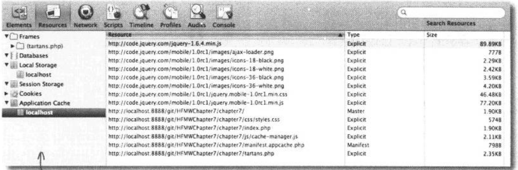
理论上讲，使用缓存清单文件的缓存管理很简单。只要列出希望离线可用的项，再为页面的<html>标记增加属性，缓存就开始了！

先别急，要知道，创建一个缓存清单并让它按我们希望的方式工作，可能有些让人糊涂，有时甚至还很烦人。有很多细节问题可能会找你的麻烦。如果能有一些好工具来帮助检查和调试到底发生了什么，这会很有好处。

开发工具来帮忙

WebKit提供了一个名为Web Inspector的工具，Chrome和Safari（基于WebKit的浏览器）都提供了这个工具。在资源（Resource）标签页中可以看到关于appCache的详细信息。

Chrome资源标签页中关于appCache的详细信息。



Safari的开发工具看起来几乎完全相同。



使用缓存清单很容易遇到麻烦。

缓存清单行为可能很难调试，而且很让人困惑，如果没有合适的工具，结构不正确的缓存清单文件将很难查找和删除。建议使用Chrome或Safari来构建和测试我们的Tartanator缓存清单。另外要注意，Internet Explorer在版本10之前不支持appCache，写这本书时，这个特性还只是处在开发人员预查阶段。

在转向移动设备之前，可以先在一个桌面浏览器上开发和测试，这会对你很有好处，不至于遇到麻烦时着急上火。

要从移动设备管理结构不正确的appCache，可能是件很痛苦的事情。

提供内容类型正确的清单



你的托管提供商或本地Web服务器很可能会支持.htaccess文件。不过这一点我们不能保证。

我们知道你能使用PHP。所以，我们会用PHP生成缓存清单文件，并用一小段代码将这个文件的content-type设置为text/cache-manifest。

Exercise

为Tartanator实现一个基本的缓存清单。

1. 将你的文件命名为manifest.appcache.php，把它放在chapter7目录中。

   扩展名必须是.php，这样我们才能执行文件中的PHP代码。

2. 在文件最上面增加下面这行代码。

```
<?php header('Content-type: text/cache-manifest'); ?>
```

   这会把这个脚本的输出结果的content-type设置为text/cache-manifest。

3. 增加一个CACHE:部分，在其中列出网站的主要页面以及JavaScript和CSS文件。

   参考285页上的语法。

4. 更新index.php、build.php、tartans.php和findevent.php的<html>标记，增加一个manifest属性。

Exercise Solution

你创建的manifest.appcache.php文件应该与下面类似。注意我们还列出了jQuery内容分发网络（CDN）的CSS和JavaScript文件。

```
<?php header('Content-type: text/cache-manifest'); ?>
CACHE MANIFEST

CACHE:
index.php
build.php
tartans.php
findevent.php
css/styles.css
js/tartanator.js
http://code.jquery.com/mobile/1.0rc1/jquery.mobile-1.0rc1.min.css
http://code.jquery.com/jquery-1.6.4.min.js
http://code.jquery.com/mobile/1.0rc1/jquery.mobile-1.0rc1.min.js
```

除了这个网站上资源的相对URL…

…还可以使用绝对URL指示你想缓存的其他域上的资源。

manifest.appcache.php

Tartanator页面上的HTML标记现在应该像这样。

```
<html manifest="manifest.appcache.php">
```

使用PHP生成清单也存在一个问题：我们无法使用期望的.appcache扩展名。





缓存清单闪亮登场

本周访谈：
让appCache听我们指挥

困惑的Web开发人员（以下简称FWD）：哎呀。我在Tartanator网站上导航时，一大堆图像、图标都没有显示，甚至即使我在线它们都不显示。

appCache：嗯，这是因为你忘了把这些资源包含在CACHE列表中。你应该把它们增加到缓存清单里。

FWD：这么说，我要列出网站上的每一个资源，否则它们就根本不会显示，是吗？

appCache：不是这样。你只需要把HTML文件需要的资源增加到缓存中。

FWD：我完全被你搞糊涂了。那么HTML文件放在缓存里有什么意义？

appCache：你已经知道把HTML文件包含在应用缓存中的一种方法：可以将它显式地列在清单文件的CACHE部分。不过，还有一点需要说明，只要HTML文件的HTML标记上有manifest属性，即使没有显式地列在缓存清单中，这些HTML文件也会包含在缓存中。只有清单中列出的页面或者manifest属性中引用了清单的页面（即在缓存中的页面）需要将其所有资源增加到缓存列表。

FWD：OK，这么说，对于放在缓存中的所有页面，如果我忘记把每一个资源都增加到CACHE部分，漏掉的那些资源就不会加载，即使我在线它们也不会加载，是吗？

appCache：嗯，也不完全正确。我想你已经了解CACHE部分，不过还没听说过NETWORK部分吧。

FWD：那又是什么？

appCache：NETWORK部分允许你定义不想缓存的资源，对于这些资源，浏览器要在连接可用时请求新的资源副本。它有一个让人不太明白的特殊通配符*，这表示未在CACHE中显式列出的所有资源都应当从服务器获取。

FWD：明白了！看来这是我想要的答案。

appCache：别那么草率，当心点。如果一个资源列在NETWORK部分中，不论是显式列出，还是通过*通配符指定，都总会请求新副本。所以你要仔细地选择路径。

要确定哪些内容离线可用，把它们列在CACHE部分。例如，看来你确实需要找出缺少的一些图标和图像。不过，对于真正动态的东西，比如登录屏幕、API调用等，可以让NETWORK通配符为你搞定。



BRAIN BARBELL

你认为Tartanator Web应用中哪些部分可能不希望离线可用？为什么？

我们需要让花格布页面和图像离线可用，所以它们要增加到CACHE部分。不过我不知道它们的URL是什么，该怎么办呢？



感觉这是个鸡生蛋还是蛋生鸡的问题，是不是？

幸运的是，我们在用PHP生成清单文件，所以可以加入一些代码来动态输出所有花格布HTML文件和当前已有图像的列表。

动手做！

在chapter7/extras文件夹中找到动态生成图像和HTML文件列表的PHP代码段。这些代码放在一个名为current_file_list.txt的文件中。

Watch it!

即使清单CACHE部分所列的资源有改变，浏览器也不会下载，除非缓存清单文件本身改变。

css/styles.css放在清单的CACHE部分中。因此，我们可以编辑styles.css，直到确定最终样式，不过浏览器不会看到这些改变。一旦把一个资源放在缓存中，只有当缓存清单文件本身改变时它才会刷新。

要对此进行管理，一种常用的方法是在清单文件中使用一个注释行，指定一个版本号。只改变这个版本号，就会让浏览器发现清单已经改变，因此会刷新下载，检查其中是否有资源是新的或者有修改。

要记住，注释行以#开头。

我们使用PHP代码自动生成当前花格布图像和HTML文件列表，这个PHP代码有一个很好的副作用，增加新花格布时会导致清单文件更新。不过，实现appCache之后，如果我们改变了其他文件，还需要让版本号递增。

Relax

我们创建的缓存清单很可能与预想的不同。

要修正有问题的appCache，在Chrome浏览器中最容易办到。在URL栏键入chrome://appcache-internals，你会看到所有当前appCache数据的有关信息。每个网站缓存都对应一个简单的Remove（删除）链接。

在Safari中，进入首选项（Preferences），选择隐私（Privacy）标签页。可以通过“Remove All Website Data”删除所有网站数据，或者使用“Details”按钮搜索有问题的网站，并删除其缓存。

在Firefox中，进入首选项（Preferences）→高级（Advanced）→网络（Network）。在离线存储（Offline Storage）部分有一个域，其中列出了已经存储数据可以离线使用的所有网站。可以在这里选择性地删除appCache。

Exercise

现在让缓存清单表现得更好一些。下面把我们学到的一些知识集成到manifest.appcache.php文件中。

1. **找出缺少的静态图标和图像。**

   jQuery Mobile使用的一些图标和图像需要增加到CACHE部分。

   如果已经在之前的步骤中发现缺少的大量图标和图像，那就太棒了！

2. **增加NETWORK:和通配符。**

   把这个代码放在CACHE:部分上面。

```
NETWORK:
*
```

3. **增加PHP代码段。**

   把extras/current_file_list.txt中的代码增加到缓存清单文件的最后。

4. **从清单中删除build.php和findevent.php。**

   不要忘记从这些页面的<html>标记删除manifest属性。

   好好想一想，这些是只有用户在线时才可用的资源。“事件”页面还什么都没有做，不过很快就会有功能了！

Exercise Solution

manifest.appcache.php现在应该是这样。

```
<?php header('Content-type: text/cache-manifest'); ?>
CACHE MANIFEST

NETWORK:
*

CACHE:
http://code.jquery.com/mobile/1.0rc1/jquery.mobile-1.0rc1.min.css
http://code.jquery.com/jquery-1.6.4.min.js
http://code.jquery.com/mobile/1.0rc1/jquery.mobile-1.0rc1.min.js
http://code.jquery.com/mobile/1.0rc1/images/ajax-loader.png
http://code.jquery.com/mobile/1.0rc1/images/icons-18-white.png
http://code.jquery.com/mobile/1.0rc1/images/icons-18-black.png
http://code.jquery.com/mobile/1.0rc1/images/icons-36-white.png
http://code.jquery.com/mobile/1.0rc1/images/icons-36-black.png
index.php
tartans.php
css/styles.css

<?php // The PHP code snippet from the current_file_list.txt file
      // goes here. We're not showing it to save space.?>
```

确保在header()行和CACHE MANIFEST行之间没有任何空行。

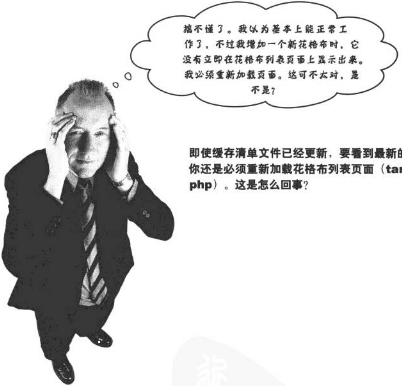
现在，保存这个文件，试试看！

Watch it!

缓存清单的一个问题可能导致整个缓存无法工作。

你还需要了解更多技巧。不过要知道，如果清单中一个资源导致404（未找到）错误，整个清单都会被忽略。类似地，一个语法错误可能导致整个清单文件失效。

你可以采取一些保护措施。可以使用有些神秘的清单验证工具（http://manifest-validator.com/），监视开发工具窗口，确保没有遗漏任何引用的资源。



即使缓存清单文件已经更新，要看到最新的内容，你还是必须重新加载花格布列表页面（tartans.php）。这是怎么回事？



Frank：我仔细研究过了。我承认对这方面很着迷。它是这样工作的。假设你的浏览器对于这个网站已经有一个缓存清单文件。另外假设你在创建新花格布之后再次访问这个花格布列表页面。浏览器会注意到缓存清单文件已经改变，它与浏览器原有的那个文件不同，因为其中的花格布文件列表已经改变。所以浏览器会立即获取新的清单文件，检查是否有更新或新增的内容。

Jim：那为什么更新或新增的这些内容没有立即显示呢？

Frank：浏览器显示页面前不会等待更新或新增的所有资源都下载。一旦浏览器完成内容下载，新增或更新的资源将准备就绪，放在浏览器的缓存中，但在页面刷新之前不会显示。

Jim：这么说，要让人们看到更新的东西，唯一的办法就是重新加载。

Frank：嗯，对于这种特定情况，我想我们可以采用一种迂回办法。我写了一个小小的JavaScript工具。

Jim：又要用JavaScript！

Frank：页面中动态的部分是由包含在页面中的PHP文件输出的，也就是list.php。它为每个花格布输出一个<li>。页面的其余部分是静态的。我们实际上只关心更新花格布列表<ul>的内容。

Jim：没错。

Frank：我的代码使用了window.applicationCache对象和它的方法来查看对缓存是否有更新。如果清单文件有改变就说明有更新，是不是？如果有更新，我们希望从list.php得到更新的输出。

Jim：对。如果缓存清单有更新，我们就知道花格布列表应该已经更新。最好不需要用户重新加载整个页面。

Frank：是啊。所以，如果缓存清单有更新，我的代码会向list.php发出一个Ajax请求。list.php不在清单中，直接请求list.php时，它总会返回表示当前花格布列表的HTML标记。然后我用list.php返回的标记替换页面上当前的（过时的）列表。就这么简单！

试一试

下面试试Frank的小JavaScript库，看看能不能帮我们渡过这个复杂的appCache难关。

1. 找到这个小代码库。

   将cache-manager.js从extras/js文件夹复制到chapter7/js。

2. 在Tartanator页面中包含这个脚本。

   把cache-manager.js脚本包含在index.php、build.php、findevent.php和tartans.php中，放在jQuery Mobile脚本后面。

```
<script src="js/cache-manager.js" type="text/javascript"></script>
```

3. 把这下面的JavaScript代码段增加到tartans.php。

   这个代码说明我们希望确保#tartans_page页面内容<div>中的#tartans-list元素得到新内容，要使用inc/list.php返回的数据。

```
<div data-role="page" id="tartans_page">
  <script type="text/javascript" charset="utf-8">
    var tartanPage = $('#tartans_page');
    tartanPage.live('pageinit', function () {
      if (!tartanPage.data.cacheManager) {
        tartanPage.data.cacheManager = new CacheManager('#tartans_page');
        tartanPage.data.cacheManager.ensureFreshContent('#tartans-list',
'inc/list.php');
      }
    });
  </script>
<div data-role="header" data-position="fixed">
```

tartans.php

there are no Dumb Questions

问：既然可以在缓存清单的NETWORK部分使用一个通配符，为什么不能在CACHE部分使用一个通配符来缓存我的网站上的所有内容，那不是更简单吗？

答：不管怎样，都不能在CACHE部分中使用通配符。这个特殊的通配符只能用在NETWORK部分中，而且它实际上不是一个通配符，不能与其他文本模式结合。这是一个独立的东西，有特殊的含义（即，对于网站上的一个给定资源，除非将它显式地列在另一部分，否则都要尽可能通过网络来请求这个资源）。

问：即使没有列在清单中，HTML标记中有manifest属性的页面也会缓存，这有什么意义？

答：这可能确实会带来混乱，让人头疼。不过关键是，这样一来，不再一次性下载和缓存给定网站上的一大堆或所有页面，而是可以采用一种“懒”方式将页面增加到缓存，也就是说，用户第一次访问页面时，才会把它们增加到缓存。

问：“一次性”是什么意思？

答：浏览器得到一个新的或更新的缓存清单文件时，它会立即开始下载所有新资源并/或检查对现有资源的更新。设计你的缓存清单时需要仔细考虑这一点。

问：我们的清单包含所有花格布图像和HTML文件。这是不是意味着一次要下载好多的资源？

答：对此我们也有点苦恼。确实有很多的HTTP请求，这一点没错。不过这些文件都很小，大多数图像都不超过1KB，HTML文件甚至更小。所以，带宽影响很小，另外，由于这些资源是异步下载的，所以对用户的影响应该微乎其微。

问：每次清单更新时浏览器都要重新下载清单中的所有资源吗？

答：谢天谢地，不会的。浏览器会查看它已经缓存的资源是否有改变，但是除非确实有修改，否则不会再次下载。

问：所有这些都还好，不过如果用户的浏览器不支持appCache会怎么样呢？

答：网站的表现会像我们增加缓存清单之前那样。它还能正常工作，不过不会提供漂亮的离线体验。

问：你完全没有提到FALLBACK部分。

答：除了CACHE和NETWORK部分，还有一个可选的FALLBACK部分，可以在这里列出在线版本不可用时要使用的离线版本—例如，使用offline.html而不是login.html。

问：appCache很不错，不过localStorage呢？感觉你只说了一半。

答：嘿，你真聪明！你已经超越我们了。第8章我们还会回来讨论localStorage（如果你没有听说过这个词，也不用担心）。

Watch it!

当心通过manifest属性实现的隐式缓存。

之前我们已经提到过，不过这里再强调一下：<html>标记上有manifest属性的页面都会缓存，而不论这个页面是否显式地列在缓存清单中。噢，对了，即使借助NETWORK部分或者试图采用任何其他技巧也无法绕过一点。

胜利（最终）属于我们

哈，真是一场浩大的战斗，不过现在终于能让Tartanator离线工作了。这说明我们已经实现了第2阶段的定制花格布部分。

- [√] 增强我们之前构建的表单，充分利用新型移动浏览器的功能。
- [√] 加入服务器端代码来处理表单，并生成用户自创花格布的相关资源（图像、HTML等）。
- [√] 确保应用这一部分的离线体验能让人接受。



祝贺你，你已经成功地控制了appCache。

编写缓存清单文件，让它按你希望的去做，这可能很难，不过你做到了！

位置感知的

实现可搜索的事件

第2阶段的另一部分是要构建一个页面，让用户搜索当地附近与花格布有关的事件。



我们想构建的查找事件（Find Events）表单。

我们想查找用户当前位置附近发生的事件。这说明，接下来需要谈谈地理定位。

地理定位如何工作

从Web浏览器实现地理定位需要使用JavaScript得到设备的当前位置。很多现代智能手机浏览器都实现了World Wide Web Consortium（W3C）地理定位API，它提供了一个简单的接口来得到地理定位数据。在这个领域还有另外一些竞争者，包括如今已经过气的Google Gears API，以及一些专用API。较老的手机（甚至一些现代智能手机）可能根本不提供浏览器中的地理定位支持。



听到地理定位时，很多人第一个想到的就是GPS（全球定位系统，Global Positioning System），不过实际上要得到一个设备的位置，还有很多其他信息来源，包括附近的WiFi网络、蜂窝信号塔、设备的IP地址、RFID（射频识别，radio frequency identification）标记，以及WiFi或蓝牙MAC（媒体访问控制器）地址。

如何向符合W3C标准的浏览器询问当前位置

实现了W3C地理定位API的浏览器可以通过navigator.geolocation对象提供地理定位。这个对象最重要的方法是getCurrentPosition，顾名思义，它的作用就是提供当前位置。

得到设备的当前位置。

```
navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
```

得到了一个位置时所要调用的JavaScript函数的函数名。

遇到一个错误时所要调用的JavaScript函数的函数名。

处理getCurrentPosition返回的信息

成功地确定一个位置时，将调用所指定的成功回调函数，并为该函数提供一个position对象。我们会得到真正关心的信息，经度和纬度，可以查看position对象coords属性的latitude和longitude来得到。

成功回调函数。

这个position对象包含我们需要的信息。

```
function successCallback (position) {
  var coords = position.coords;
  alert(coords.latitude + ',' + coords.longitude);
}
```

position最重要的属性是coords对象……

……coords对象最重要的属性则是latitude和longitude。

错误回调函数。如果出了问题就会调用这个函数。

这两个函数只会弹出一个JavaScript警告框（有时还有些烦人）。

```
function errorCallback(error) {
 alert(error.message);
}
```



地理定位JavaScript磁贴

把代码磁贴放在下面地理定位JavaScript中正确的位置。每个磁贴只能使用一次，不过可能有些磁贴根本用不到。

message
navigator.geolocation
longitude
coords
coords
position
onGeoSuccess
"Browser doesn't support geolocation"
position
position
getCurrentPosition()
getCurrentPosition
onGeoError
getLocation
latitude
coordinates

```
function getLocation () {
  // Verify the browser supports W3C geolocation
  if (..............................) {
    navigator.geolocation. .................... (................, onGeoError);
  } else { // It doesn't
    onGeoError(new Error(....................................));
  }
}

function onGeoSuccess (................) {
  var.................... = ................coords;
  alert(coordinates.................+ ',' + coordinates...............);
}

function ..................(error) {
  alert(error...................);
}
......................();
```

答案见第302页。

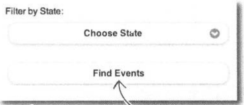
查找事件页面起步：基准

查找事件（Find　Events）页面的想法很明确，就是要查找事件。现在先看看如何在可用的浏览器中使用JavaScript查找用户的位置。不过我们还是采用先创建基准体验的做法。如果一个用户的浏览器不支持JavaScript或地理定位呢？

下面先从基本的开始，创建一个事件搜索表单，这里不需要JavaScript或地理定位。在一个选择列表中提供美国各州的一个简单列表，这就是我们的默认基准。

一个非常简单的表单，可以按州搜索事件。

这是一个名字和ID为state_filter的select元素。



事件数据库只有美国的事件……至少现在是这样。

Exercise

在findevent.php上建立一个简单的事件搜索表单作为我们的基准。把表单放在页面内容<div>中。

1. 为这个表单指定一个GET方法，另外id为search_form。将action属性设置为events.php。

   稍后就会告诉你在哪里找events.php。

2. 增加一个select输入元素，要显示美国50个州的名字作为选项，这些值应当是各个州的两字母邮政编写。

   在extras/events目录中找到us_states.txt，这个文件会对你很有帮助。

3. 提交输入按钮id应为search_submit。

   不要忘记把这些域包围在一个<div>中，其data-role为fieldcontain（jQM方式）。



地理定位JavaScript磁贴答案

这些磁贴的位置放对了吗？

```
function getLocation () {
  // Verify the browser supports W3C geolocation
  if ( navigator.geolocation ) {
    navigator.geolocation. getCurrentPosition ( onGeoSuccess , onGeoError);
  } else { // It doesn't
    onGeoError(new Error( "Browser doesn't support geolocation" ));
  }
}

function onGeoSuccess ( position ) {
  var coordinates = position .coords;
  alert(coordinates. latitude + ',' + coordinates. longitude );
}

function onGeoError (error) {
  alert(error. message );
}
getLocation ();
```

构造这个代码之后，把它放在一个名为geolocation.js的文件中，这个文件在chapter7/js/目录中。然后在findevent.php页面上包含这个文件，为此要在data-role="page"的<div>中使用一个<script>标记。

这应该放在搜索表单上面。

不需要这些磁贴。

coords　position

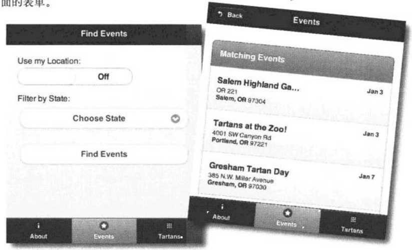
getCurrentPosition()　coords

集成地理定位

现在我们有了一个基本表单。噢，还有一些烦人的JavaScript警告框。

下面这一部分很有意思，为支持地理定位的浏览器加入地理定位功能。完成以下修改之后，对于支持地理定位的浏览器，会显示类似下面的表单。

“装扮”好这个表单后，如果用户使用支持地理定位的浏览器，他们就能利用当前的位置查找事件。



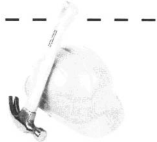
Exercise Solution

你的基本表单如下所示。

```
<div data-role="content">
  <script type="text/javascript" src="js/geolocation.js"></script>
  <form method="get" action="events.php" id="search_form">
    <div data-role="fieldcontain">
      <label for="state_filter">Search by State</label>
      <select id="state_filter" name="state_filter">
        <option value="">Choose State</option>
        <option value="AL">Alabama</option>
        <!-- ETC -->
      </select>
    </div>
    <div data-role="fieldcontain">
      <input type="submit" value="Find Events" id="search_submit" />
    </div>
  </form>
</div><!-- /content -->
```

地理定位构造

我们要“装扮”geolocation.js中的代码。下面对其中的重点做详细分析。



```
(function () {
  var $page, $searchForm, $submitButton, $stateFilter;

  $page = $('#event_page');
  if (!$page.data.initialized) {
    $page.live('pagecreate', initGeo);
    $page.data.initialized = true;
  }

  function initGeo() {
    $searchForm = $('#search_form');
    $submitButton = $('#search_submit');
    $stateFilter = $('#state_filter');
    if (navigator.geolocation) {
      initGeoOptions();
    }
  }
  function initGeoOptions() {
    var $latField, $longField, $flipSwitch;
    $flipSwitch = $('<select name="usegeo" id="usegeo"
data-role="slider"><option value="off">Off</
off><option value="on">On</option></select>').
change(toggleLocation);
    $flipSwitch.prependTo($searchForm).wrap('<div data-
role="fieldcontain"></div>');
    $flipSwitch.before('<label for="usegeo">Use my
Location:</label>');
    $latField = $('<input type="hidden" />').attr({ name
: 'latitude', id : 'latitude'})
    $longField = $('<input type="hidden" />').attr({
name: 'longitude', id : 'longitude' })
    $latField.appendTo($searchForm);
    $longField.appendTo($searchForm);
  }
```

页面初始化时，调用initGeo函数。

如果浏览器支持地理定位，初始化地理定位相关的表单元素。

增加一个滑动开关，允许用户使用他的当前位置。

增加两个隐藏域，保存由地理定位得到的经度和纬度值。

toggleLocation绑定到滑动开关的change事件。

```
  function toggleLocation(event) {
    var geoActivated = ($(event.target).val()
== 'on') ? true : false;
    if (geoActivated) {
      $submitButton.button('disable');
      $stateFilter.selectmenu('disable');
      $.mobile.showPageLoadingMsg();
      navigator.geolocation.getCurrentPosition(
onGeoSuccess, onGeoError);
    } else {
      $stateFilter.selectmenu('enable');
      $submitButton.button('enable');
    }
  }
  function onGeoSuccess(position) {
    var coordinates = position.coords;
    $('#latitude').val(coordinates.latitude);
    $('#longitude').val(coordinates.longitude);
    $.mobile.hidePageLoadingMsg();
    $submitButton.button('enable');
  }
  function onGeoError(error) {
    $('#usegeo').val('off').trigger('change');
    alert(error.message);
  }
})();
```

如果用户将它打开，会禁用州过滤器，并禁用提交按钮（地理定位完成时会重新启用）。

在这里触发地理定位。

关闭时，重新启用按州过滤功能。

这里是我们的成功回调。我们已经做了修改，使用用户的位置数据来更新经度和纬度隐藏域。

出现错误时，把“使用地理定位”的滑动开关滑动到关闭位置，触发其change事件（这个事件绑定到上面的toggleLocation）。

试试看！进入表单，感觉一下这个新JavaScript会为我们做什么。

在chapter7文件夹的extras/js/enhanced_geo_form.js中可以找到这个代码。

更新geolocation.js文件，保存，然后尝试更新后的“查找事件”页面。



Jim：我刚才在一些设备上检查了地理定位部分。我知道BlackBerry OS 5浏览器支持地理定位，但是并没有显示“Use My Location”（使用我的位置）开关。

Frank：对。BlackBerry OS 5浏览器确实支持地理定位，不过它可不是W3C风格。我正在查这个问题。另外较老的Android浏览器也没有W3C风格的地理定位。它们都基于Google Gears。

Jim：听上去很复杂。

Frank：确实是。另外我们时间很紧。我觉得我们没时间这样一步一步研究每一个细节，对一大堆不明确的设备挨个测试。我发现一个开源的JavaScript库，很方便的，可以在大量不同平台上（不论是否符合W3C）处理地理定位。它的名字很简单，就叫做geo-location-javascript。

Jim：不过，这是不是意味着我们必须重构现有的所有JavaScript代码？

Frank：不用，这个库的API模仿了W3C规范的风格。所有主要方法和属性名都是一样的。我们只需要修改几行代码，然后就不用那么担心跨平台兼容性问题了。

Jim：噢，既然谈到对JavaScript的编辑……地理定位失败时弹出的错误真是很难看，而且最好措辞有点技巧。

Frank：对。我想我们应该使用一个jQuery Mobile对话框，比如花格布创作表单上为表单错误建立的那些对话框。

Jim：啊，你还没告诉我呢！

Frank：如果你想提交一个花格布表单，但是没有指定标题，或者如果在增加颜色-大小组合域时忘记了颜色或大小，就会弹出一个对话框，警告你有错误。它会采用一种漂亮的jQM样式。

Jim：怎么建立jQuery Mobile对话框呢？

Frank：jQuery Mobile对话框是独立的jQM页面。可以查看dialogs目录中的对话框页面，然后通过JavaScript显示。

Jim：嗯，很酷。我大概明白了。

Frank：OK。你一定行的。接下来我要加入这个地理定位库了。

动手做！

把extras/js/geo.js文件复制到chapter7/js文件夹。这就是geo-location-javascript库。如果你想了解更多，还可以访问这个项目的主页http://bit.ly/sx7JrH。

Exercise

更新“查找事件”页面和地理定位JavaScript，使用这个跨平台地理定位代码，并在一个弹出的对话框中显示地理定位错误。

1 更新findevent.php，包含这个新地理定位库的JavaScript。

```
<div data-role="content">
<script src="http://code.google.com/apis/gears/gears_init.js" type="text/
javascript" charset="utf-8"></script>
<script src="js/geo.js" type="text/javascript" charset="utf-8"></script>
<script type="text/javascript" src="js/geolocation.js"></script>
```

2 更新geolocation.js，使用这个新库。

找到并修改以下代码：

```
if (navigator.geolocation) {
    initGeoOptions();
}
```

修改为：

```
if (geo_position_js.init()) {
  initGeoOptions();
}
```

并修改以下代码：

```
navigator.geolocation.getCurrentPosition(onGeoSuccess, onGeoError);
```

修改为：

```
geo_position_js.getCurrentPosition(onGeoSuccess, onGeoError);
```

3 为地理定位错误创建并调用一个对话框。

依照dialogs目录中的其他页面，为地理定位错误创建一个措辞适当的错误对话框。把这个文件命名为geolocation_error.html。

接下来，用这样一个弹出对话框替换onGeoError函数中的警告框。在js/tartanator.js文件中可以找到弹出对话框的例子。应该只需要调整URL参数。

Exercise Solution

出现错误时，我们把基本的JavaScript警告框转换为一个jQuery Mobile对话框，现在你的onGeoError函数应该如下所示。相信你已经为对话框本身建立了HTML页面！

```
function onGeoError(error) {
  $('#usegeo').val('off').trigger('change');
  $.mobile.changePage( "dialogs/geolocation_error.html", {
    transition: "pop",
    reverse: false,
    role: 'dialog'
  });
}
```

Relax

我们提供了一些精巧的JavaScript。

我们承认时间很紧张，有些代码还没有做深入的解释。不过这不是一本《Head First JavaScript》，我们不希望你分心，应该把重点放在当前要解决的主要任务上，也就是要控制住appCache，并充分利用PhoneGap的mediaCapture API（甚至包括还不提供支持的浏览器）。

试一试

可以试一试了！下面来看它的实际使用。关联搜索代码和示例数据不需要花费太大功夫。

1. **从extras/events目录复制所需的文件。**

   把event_search.inc和event_list.inc复制到chapter7/inc目录。把events.php复制到chapter7目录。

2. **试试看！**

   在你选择的移动浏览器中进入“查找事件”页面，点击“Use My Location”（使用我的位置）开关。

什么也没找到

哈，真是抱歉。除非你很幸运，刚好在美国俄勒冈州波特兰的中心区，否则第一次基于位置搜索时可能都会得到这个结果：



除非你靠近波特兰，否则你第一次尝试时会得到这个结果。

这是因为所有示例数据都是本地的，搜索半径设置为25英里。也许该增加一些你自己的虚构事件，让它能提供一些具体结果。

Exercise

在示例数据中增加一些你自己虚构的事件。找到chapter7/inc/中的event_list.inc 文件。其中有一个很大的PHP数组，这就是表示事件的虚构数据。向这个数组增加你自己的一些元素（在你周围25英里范围内），各元素包括一个地址和纬度/经度对。

如果你对PHP很不熟悉，可以试着直接更新现有数组项的地址和纬度/经度对。纬度、经度和州就是搜索脚本查找匹配所用的全部信息。

使用下面这个工具可以很快得到任何地址的纬度/经度对：http://itouchmap.com/latlong.html。

看起来太棒了！这正是我刚开始为花格布爱好者构思移动Web应用时所设想的效果。



真不错！我们已经成功地集成了多个特性：渐进增强、离线模式和位置感知，得到了一个超级移动Web应用。



BRAIN POWER

更上一层楼：你能想到如何增加一个range输入域允许用户确定搜索半径吗？它应该在什么时候显示？实际上，如果你能确定如何增加一个名为radius的输入域，后台搜索脚本已经做好配置，可以使用这个输入域来调整搜索半径！



BULLET POINTS

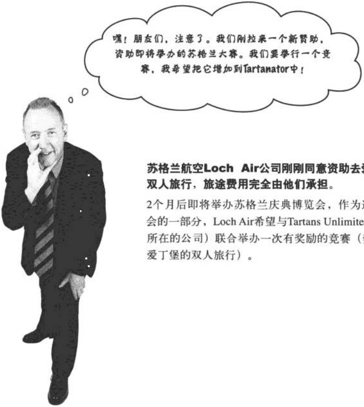
- 在移动设备上感觉自然舒适的Web应用通常都有一些共同的特点，如周全的渐进增强、完善的离线模式和位置感知特性。
- 从基准开始，这是一种很好的方法，可以支持尽可能多的用户，然后可以为功能更强的移动浏览器提供周全的渐进增强，得到让人惊艳的效果。
- 建立合适的离线体验很重要：不能指望移动设备总有可用的数据连接。
- 我们可以利用浏览器中的appCache特性，创建一个缓存清单文件来指示浏览器哪些资源要缓存从而离线可用。
- 缓存清单的构造表面上看很简单，但实际上有很多问题需要仔细考虑。
- 在缓存清单文件中，可以（在CACHE部分）指示哪些资源要下载并缓存，从而离线可用，另外（在NETWORK部分）指示哪些资源要在互联网连接可用时请求刷新版本。
- 浏览器地理定位在很多现代智能手机中都得到了广泛支持。很多浏览器实现了W3C地理定位API，它为JavaScript提供了navigator.geolocation对象。
- 对于支持相应特性的浏览器，我们增强了花格布创作表单，类似地，还可以增强查找事件表单来集成地理定位（如果支持地理定位）。
- 某些移动浏览器（特别是在比较老的智能手机上）尽管有地理定位实现，但不是W3C风格。要处理这些特殊情况，我们使用了一个第三方地理定位库，名为geo-location-javascript。
- geo-location-javascript库模拟了W3C API，只需对我们原来的W3C特定的JavaScript代码稍做改动。

花格布搜寻：实现原生



有时候你得实现原生。这可能是因为你需要访问移动浏览器（还）没有提供的某些功能。或者，也许是你的客户要求有App Store中的一个应用。我们期盼着将来能够在浏览器中访问我们想要的一切，希望移动Web应用同样拥有那些诱人的原生应用才拥有的特性。这种情况下，我们可以选择混合型开发，继续使用Web标准编写代码，再使用一个库在我们的代码与设备原生能力之间搭建桥梁。基于Web技术构建的跨平台原生应用？不会是一个糟糕的折中吧，嗯？

机会又来了



苏格兰航空Loch Air公司刚刚同意资助去爱丁堡的双人旅行，旅途费用完全由他们承担。

2个月后即将举办苏格兰庆典博览会，作为这个博览会的一部分，Loch Air希望与Tartans Unlimited（Ewan所在的公司）联合举办一次有奖励的竞赛（奖励就是爱丁堡的双人旅行）。



花格布搜寻（Tartan Hunt）竞赛要求参赛者搜寻隐藏在某些重要赞助商摊位上的花格布。

参赛者要努力找到花格布，并用他们设备上的相机为所列的各个花格布拍照，找到的花格布再与隐藏的花格布列表核对。一旦一个参赛者找到了所有花格布，他就能显示完整的花格布列表，赢得这次旅行。



很遗憾，这里有一个问题。

对于大多数移动浏览器来说，并不能访问相机（以及其他一些设备特性，比如录音、文件系统操作、网络状态和通信录数据等等）。



很多设备能力（如相机）无法从移动浏览器访问。

现在情况在慢慢好转，不过仍然有很多特性我们无法从浏览器访问。

Frank：显然我们没有时间、也没有相应的预算针对多种平台将Tartanator重写为一个原生应用，并增加Ewan希望的新特性。

Jim：这是不是说我们没办法完成这个工作了？那可太糟糕了。

Joe：我想这里能不能做个折中。我们可以考虑提交一个混合型应用。

Jim：那又是什么？

Joe：混合型应用是用Web技术编写的原生应用。由于它是原生的，所以我们可以得到类似相机等设备能力，不过代码本身用标准Web编写，所以我们可以在多个平台上共享代码。

Frank：不过，既然这是基于Web的，那些特性在浏览器中不可用，那该怎么办呢？

Joe：有很多框架和平台可以在Web应用和原生代码之间搭建桥梁。它们对我们的Web应用做一些处理，然后为不同的平台提供打包应用（原生应用）。这些框架和平台会帮我们处理各种细节问题，这样我们就可以只使用我们知道的Web标准。

Jim：那么，是不是要把整个Tartanator转换成这样一个混合型应用？

Joe：我觉得我们没时间把整个应用变成原生应用并有效地进行测试。按现在的时间进度要求，能不能只为这次竞赛创建一个原生应用，你觉得这个建议怎么样？

Frank：这个主意不错。这样我们就不用冒着风险急急忙忙地修改Tartanator Web应用。这有一个好处—参赛者可以下载一个只针对这个竞赛的特殊应用。Joe，如果你认为这个混合型应用可以在我的平台上使用，下面就跟Ewan说一说这个想法吧。

混合型应用如何工作？

可以使用标准web技术编写你的代码，如HTML5、CSS和JavaScript，这与我们以前构建网站完全相同。然后利用一个框架或平台作为桥梁，它会提供一个公共API，可以用它以原生方式访问不同设备平台上的特性。这个框架或平台填补了JavaScript代码和设备原生代码（对于所支持的各个平台）之间的缺口，并为各个平台生成应用。



用PhoneGap填补Web-原生缺口

PhoneGap是一个开源的HTML5平台，允许你使用我们之前讨论的混合型方法为多个移动操作系统创建原生应用。你可以按你习惯的方式使用HTML、CSS和JavaScript编写代码。PhoneGap通过一个一致的、跨平台的JavaScript API在你的代码和原生代码之间搭建桥梁。

PhoneGap由总部在温哥华的Nitobi公司开发，这家公司于2011年10月被Adobe收购。不过，不用担心，Adobe已经采取了措施确保PhoneGap仍然保持开源，它将PhoneGap授权给非盈利的Apache软件基金会（Apache Software Foundation）。新创立的Apache Cordova项目会确保PhoneGap代码基一直保持开源。

PhoneGap Build

PhoneGap Build是一个基于Web的服务，允许你同时为多个平台进行编译。利用这个服务，可以跳过一些组件的安装，至少是一些SDK、插件、IDE以及你想要支持的各个平台的某些组件，从而消除构建过程的一些麻烦。

PhoneGap支持多达最多7个平台，不过这一章将使用PhoneGap Build，它只支持Apple iOS、Android、BlackBerry、WebOS和Symbian。

利用标准PhoneGap开发时，你要自己完成编译和构建，而使用PhoneGap Build则不同，你只需上传一个zip文件（或者从一个代码存储库取得），它会完成余下的工作。



我们要向你展示如何使用PhoneGap Build构建一个混合型Android应用。

如果你没有Android设备，不用担心。我们会告诉你如何为真正的设备以及模拟的虚拟设备构建和安装应用。



我们选择Android作为示例平台是有原因的。

一方面，这是一个流行的平台（确实如此）。不过，另外还有一个原因，在设备上构建和部署iOS应用有一个条件。要求必须是iOS开发计划的成员（每年的会费是99美元）。我们不希望让你多花钱。

Android还提供了一个免费的SDK，支持在不同版本的Android OS上模拟多种虚拟设备。另外，Android SDK在很多平台上都能运行。而倘若使用Xcode（这是iOS开发所需的开发工具包），则要求必须有Mac平台，否则情况可不妙。

你的开发环境准备好了吗？



如果你还没有准备好开发环境，赶快看看附录iv。你需要安装合适的组件，而且要了解如何在虚拟和真实设备上安装和卸载应用，然后再来看这一章后面的部分。



JavaScript先行。

要完成这一章，并不要求你是一个JavaScript高手，不过有一点不得不承认，如今的Web应用开发总少不了JavaScript。

要使用PhoneGap与原生功能通信（如相机API），我们需要使用一种客户端技术，JavaScript显然是完成这个任务的最佳选择。

there are no Dumb Questions

问：构建混合型应用还有其他工具、产品或服务吗？

答： 当然有，其中有两个最知名的混合型开发工具：Appcelerator Titanium和即将推出的Sencha Touch 2产品。

问：PhoneGap支持哪些平台？

答： PhoneGap目前支持Android、Bada、BlackBerry、iOS、Symbian、WebOS和Windows Phone 7。可以查看http://phonegap.com/about/features，其中提供了一个列表，列出了哪个平台的哪个版本上支持哪些特性。

问：PhoneGap Build与PhoneGap支持同样的平台吗？

答： 不完全是。写这本书时，PhoneGap Build还不支持Windows Phone 7或Bada。

问：PhoneGap免费吗？

答： 是的。PhoneGap（也就是现在的Apache Cordova）是开源而且免费的。

问：PhoneGap Build呢？它也免费吗？

答： 这要看情况。PhoneGap Build提供了分级服务。它的免费版本允许开发人员有一个私有应用，另外可以有任意多个公共应用。我们将使用这个免费服务来构建Tartan Hunt应用。PhoneGap Build其他级别的服务会增加私有应用和项目合作者的个数。

问：为什么必须安装Android SDK？

答： 我们不会编译或构建Tartan Hunt应用，这个工作将由PhoneGap Build为你完成。不过我们还需要在一个模拟器或设备（或者这二者）之上完成安装和测试。为此，我们需要SDK提供的一些工具。

不过，不需要安装Eclipse IDE，但倘若我们自己开发和构建代码往往就需要安装这个IDE。所以至少我们可以少几个步骤！

问：为什么不能直接在Web浏览器中访问这些特性呢？

答： 在移动设备上，有些工作不能从浏览器完成，这是有一定原因的。

其中需要考虑的一个因素是安全性。要允许浏览器访问类似文件系统或用户通信录之类的特性，这要特别小心，必须仔细考虑如何把它们适当地放在沙箱里，而要想妥善地做到这一点是需要花时间的。

另外，为这些特性提出的标准可能是新的或者不完备的，或者可能有多个版本可以采用。由于这些问题的存在，可以想见我们前进的道路上肯定有坎坷，浏览器提供的支持可能会有冲突或不一致。

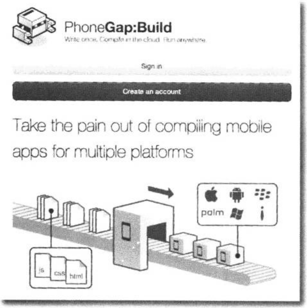
不过，未来总会实现标准化，能够提供各种设备的API！下面就来享受我们的旅程吧！

BRAIN POWER

如果你使用的不是Android设备，但确实是PhoneGap Build支持的平台之一，或者如果你有一个Apple开发账户，建议你可以试着在你的平台上构建和测试本章中的例子。

我们的测试组已经在多种版本的Android和iOS上成功地运行了Tartan Hunt应用。

熟悉PhoneGap Build



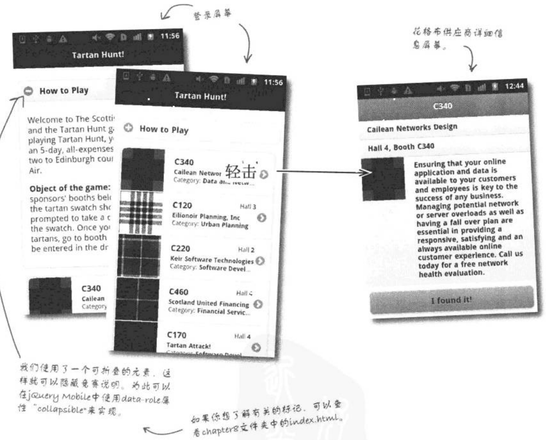
动手做！

访问https://build.phonegap.com，并创建一个账户。这是免费的，而且创建账户会很快，不会花费太长时间。

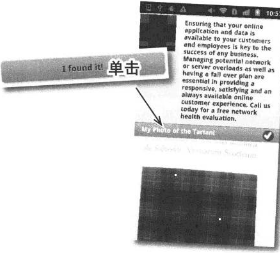


应用如何工作？

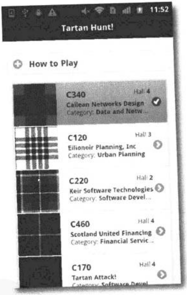
应用的登录页面会显示一个竞赛说明和要查找的一个花格布列表。单击列表中的一项就会显示可以在哪个供应商的摊位上找到这个花格布，并显示这个供应商的更多信息。



跟踪已经找到的花格布



在最终完成的应用中，单击“I found it!”（我找到了！）按钮，会启动设备的相机，参赛者可以拍下他们找到的花格布。照片将显示在详细页面上，另外在登录页面的列表中相应的花格布会标记为“已找到”。



我们已经有了基本HTML布局、CSS和图像。现在要让它们发挥作用，使用PhoneGap Build构建一个原生Android应用。

实现步骤

- ☐ 建立并配置一个PhoneGap Build项目。打包压缩当前的HTML、CSS和图像。构建应用，并在Android设备或模拟器上安装这个应用。
- ☐ 增加相应功能，允许参赛者标记他们已经找到的花格布。
- ☐ 增加相应功能，允许参赛者保存他们找到的花格布的照片。

Tartan Hunt项目剖析

下面是index.html中内容<div>的简化结构。

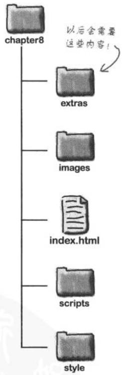
内容结构是一个<ul>，其中包括很多嵌套<ul>。

```
<div data-role="content">
  <div data-role="collapsible">...</div>
  <ul id="vendors">
    <li>
      <ul class="details">
        <li>...</li>
        <li>...</li>
        ...
      </ul>
    </li>
    <li>
      <ul class="details">
        <li>...</li>
        <li>...</li>
        ...
      </ul>
    </li>
    ...
  </ul>
</div>
```

index.html

每个花格布<li>元素中有一个嵌套列表(ul.details)，其中包含有关供应商的更多信息和"I found it!"按钮。

ul#vendors中对应每个要搜寻的花格布分别有一项（<li>）。



这个项目中我们又要使用jQuery Mobile，所以整体看起来与Tartanator基本上是一致的。

我们已经基本上为PhoneGap Build做好准备

作为起点，现在结构基本上已经准备好，可以放入PhoneGap Build了。

PhoneGap Build项目的结构要基于W3C Web部件规范。Web部件是封装的Web应用，可以作为独立的客户应用。如果采用最基本的形式，Web部件至少包括一个起始文件（在这里就是index.html）和一个配置文件（XML文件）。可以在包中增加应用所需的任意多个文件，如图像、CSS、JavaScript等。

想看看规范？可以在这里找到：www.w3.org/TR/widgets/。

我们基本上已经准备好了。不过还需要创建一个配置文件，为PhoneGap Build提供有关应用的一些必要的详细信息。

PhoneGap Build会在我们的项目中查找一个名为<u>config.xml</u>的文件。下面就来建立这个文件！

Exercise

下面建立我们的第一个PhoneGap Build应用。首先，我们需要一个配置文件。在chapter8目录中创建一个名为config.xml的文件。使用下面的模板完成这个文件的内容。

为应用指定人可读的名字和描述。

```
<?xml version="1.0" encoding="UTF-8"?>
<widget xmlns      = "http://www.w3.org/ns/widgets"
      xmlns:gap    = "http://phonegap.com/ns/1.0"
      id           = ""
      version      = "1.0.0">
  <name></name>
  <description></description>
  <author href=""
      email="">YOUR NAME HERE
  </author>
  <icon src="images/touch-icon-iphone4.png" height="114" width="114" />
</widget>
```

下面使用com.hfmw.tartanhunt作为我们的应用ID。

你的网站和email地址。如果你没有自己的网站，可以使用 http://hf-mw.com。

在这里增加图标！这些图标会在用户主屏幕上显示。我们已经为你增加了第一个图标。可以在images目录中找到其余的图标，把它们也加上。

config.xml

要记住，PhoneGap Build应用可以从一个zip文件构建，也可以从代码存储库获取。我们将采用从zip文件构建的方法。

打包压缩chapter8目录的整个内容。最后的文件叫什么并不重要，只要它是一个zip压缩文件（有一个.zip扩展名）。

Exercise Solution

你的config.xml文件应该与下面类似。不过别忘了你的标识！希望你使用自己的名字和内容。

```
<?xml version="1.0" encoding="UTF-8"?>
<widget xmlns      = "http://www.w3.org/ns/widgets"
      xmlns:gap    = "http://phonegap.com/ns/1.0"
      id           = "com.hfmw.tartanhunt"
      version      = "1.0.0">
  <name>Tartan Hunt</name>
  <description>A companion application for the Scottish Celebration:
win a trip for 2 to Edinburgh from Loch Air!</description>
  <author href="http://www.hf-mw.com"
      email="help@hf-mw.com">
      Lyza Gardner and Jason Grigsby
  </author>
<icon src="images/touch-icon-iphone4.png" height="114" width="114" />
<icon src="images/touch-icon-ipad.png" height="72" width="72" />
<icon src="images/touch-icon-iphone.png" height="57" width="57" />
<icon src="images/icon-48.png" height="48" width="48" />
<icon src="images/icon-36.png" height="36" width="36" />
</widget>
```

PhoneGap Build会为设备提供适当的图标……

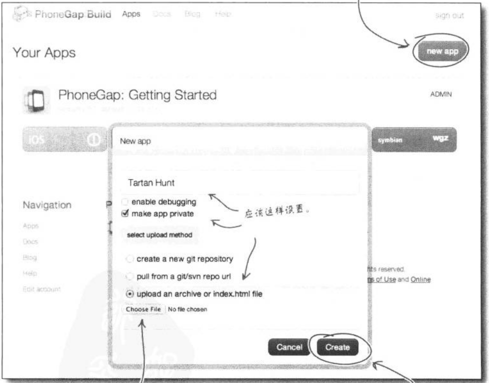
config.xml

Exercise

登录PhoneGap Build，查看控制面板。现在该创建我们的应用了！

https://build.phonegap.com

单击new app（新建应用）按钮，开始创建应用。



应该这样设置。

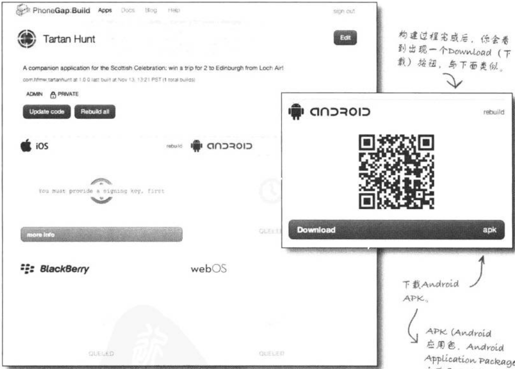
从chapter8目录选择你刚才创建的zip文件。

单击Create来创建你的应用。

下载应用

上传zip文件之后，PhoneGap Build将你的应用排队，等待构建。等一两分钟之后，刷新页面。你会看到对应已成功构建这个应用的各个平台分别有一个Download（下载）按钮。我们想下载Android包(APK)文件。

构建过程完成后，你会看到出现一个Download（下载）按钮，与下面类似。



下载Android APK。

APK (Android应用包，Android Application Package)文件是可以安装的打包应用。

OK，下面就来安装这个应用。

选择你的路

在模拟器上安装

要在一个虚拟设备上安装这个应用，启动Android SDK。进入Tools（工具）→Manage AVDs（管理AVD），会显示已安装的Android虚拟设备（Android virtual Devices，AVD）列表。对应一个比较新的Android版本启动一个设备。安装应用时这个模拟设备必须已经运行。除非模拟器已经运行，否则不要试图安装应用。

或

在真实设备上安装

如果你有一个真实的Android设备，太好了！把你的Android设备连接到计算机的USB口。对，就这么简单。

准备，瞄准，开火！

不论在虚拟设备上还是真实设备上部署，安装应用的过程都是一样的。打开一个终端窗口（Mac或Linux），或者在Start（开始）菜单的Search（搜索）框中键入cmd（Windows），得到一个命令行窗口，再使用cd命令切换到下载Android APK文件的目录。

安装应用的命令如下：

```
adb install <包文件名>
```

稍后会在附录iv中更详细地介绍这个命令。

别忘了！运行这个命令之前，你的（真实）设备必须连接到计算机，或者必须完全启动模拟器。

我们的APK文件名为TartanHunt.apk。

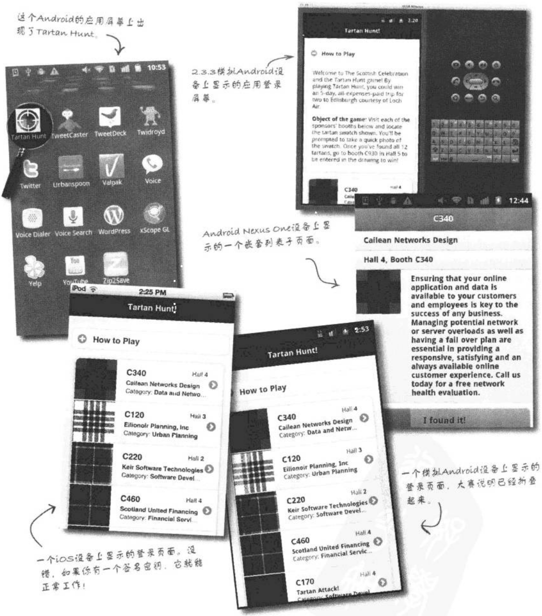
```
File Edit Window Help Tartans
$ adb install TartanHunt.apk
1157 KB/s (683431 bytes in 0.576s)
      pkg: /data/local/tmp/TartanHunt.apk
Success
$
```

应该能看到这样的结果……

建议使用Android 2.3或4.0+的AVD。之前版本的模拟器有些问题，特别是在（我们接下来要使用的）相机方面会有麻烦。

现在应该能够在设备或模拟器的应用屏幕上看到Tartan Hunt应用了。启动这个应用！

第一次运行这个应用时，启动可能会花点时间——最多30秒。


这个Android的应用屏幕上出现了Tartan Hunt。
2.3.3模拟Android设备上显示的应用登录屏幕。
Android Nexus One设备上显示的一个嵌套列表子页面。
一个模拟Android设备上显示的登录页面，大赛说明已经折叠起来。
一个iOS设备上显示的登录页面。没错，如果你有一个签名密钥，它就能正常工作！
Tartan Hunt
TweetCaster
TweetDeck
Twidroyd
Twitter
Urbanspoon
Valpak
Voice
Voice Dialer
Voice Search
WordPress
xScope GL
Yelp
YouTube
Zip2Save
Tartan Hunt!
How to Play
Welcome to The Scottish Celebration and the Tartan Hunt game! By playing Tartan Hunt, you could win an 5-day, all-expenses-paid trip for two to Edinburgh courtesy of Loch Air.
Object of the game: Visit each of the sponsors' booths below and locate the tartan swatch shown. You'll be prompted to take a quick photo of the swatch. Once you've found all 12 tartans, go to booth C930 in Hall 5 to be entered in the drawing to win!
C340
Cailean Networks Design
Hall 4, Booth C340
Ensuring that your online application and data is available to your customers and employees is key to the success of any business. Managing potential network or server overloads as well as having a fail over plan are essential in providing a responsive, satisfying and an always available online customer experience. Call us today for a free network health evaluation.
I found it!
C340 Hall 4 Cailean Networks Design Category: Data and Netwo...
C120 Hall 3 Eilionoir Planning, Inc Category: Urban Planning
C220 Hall 2 Keir Software Technologies Category: Software Devel...
C460 Hall 4 Scotland United Financing Category: Financial Servi...
C170 Hall 4 Tartan Attack!



有意思。我们只有一个HTML文件。不过点到每个花格布时都会带我去一个不同的页面。

jQuery Mobile会把嵌套<ul>自动转换到单独的“页面”。

它把各个嵌套列表转换为一个<div>，这个<div>的data-role为page，点击列表项时会使用这个<div>作为活动“页面”。理论上讲实际确实只有一个页面。另外可以注意到，父列表元素的标题（在这里就是供应商摊位的编号）会作为嵌套列表页面的标题。

嘿，那么BlackBerry呢？

如果在PhoneGap Build上访问控制面板，可能会看到这样的界面：



通过提供一个签名密钥（如果有的话），就可以修正iOS状态。

BlackBerry的苦脸是怎么回事？嗯，很遗憾，目前PhoneGap Build、BlackBerry和包含连字符的文件名结合使用时还存在问题。我们要用到的很多jQuery Mobile文件的文件名中都有连字符。

BRAIN POWER

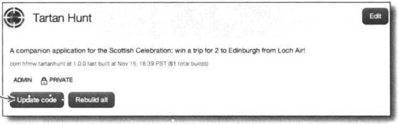
为了让PhoneGap Build Tartan Hunt应用在BlackBerry OS上工作，需要调整jQuery Mobile的图标文件名，并更新jQM源代码中对这些图标文件的引用。尽管不会花费太长时间，不过这意味着会“改变”jQuery Mobile的内核，带来维护负担。你觉得呢？这样值得吗？

为简洁起见，另外由于Ewan已经明确表示没有信心支持BlackBerry平台，所以这里我们不打算解决这个问题……

Exercise

下面增加一个开始屏幕启动画面，这样应用加载时用户就不用总看着一个乏味的空屏幕了。使用PhoneGap Build config.xml文档（位于https://build.phonegap.com/docs/config-xml）来确定如何做到。找到chapter8/extras/images目录中的两个启动画面图像，把它们移到chapter8/images文件夹。

更新config.xml，再压缩chapter8，在PhoneGap Build上重新构建这个应用，再重新安装，可以看到新的启动画面会起作用。



找到“Update code”按钮，上传你更新后的zip文件。上传之后就会自动开始重新构建过程。

答案见第334页。

Watch it!

完成修改之后，在重新安装之前需要卸载原来的应用。

每次在PhoneGap Build上重新构建应用并且需要在一个设备或模拟器上重新安装时，需要首先将它卸载。如果设备通过USB与计算机连接，或者模拟器（要从这个模拟器中卸载应用）正在运行，此时可以运行这个命令。

```
File Edit Window Help TartanHunting
$ adb uninstall com.hfmw.tartanhunt
Success
$
```

需要注意，运行uninstall命令要使用应用的包ID（com.hfmw.tartanhunt或config.xml文件中设置的ID），这与install命令不同，install命令中要使用APK文件名。

不错，漂亮！

太棒了。我们已经有了一个应用。现在让它完成它该做的事情。我们把功能部分分为两步实现。

首先，我们希望应用能够记住用户找到了哪些花格布。参赛者点击“I found it!”按钮时，需要应用把它记录下来。



祝贺你，你刚刚构建了一个原生移动应用。

不太难，是不是？你可以使用你的Web开发技术构建原生应用，一点都不费劲！

存储用户已经找到哪些花格布

- [x] 建立并配置一个PhoneGap Build项目。打包压缩当前的HTML、CSS和图像；构建应用，并在Android设备或模拟器上安装这个应用。
- [ ] **增加相应功能，允许参赛者标记他们已经找到的花格布。**
- [ ] 增加相应功能，允许参赛者保存他们找到的花格布的照片。

首先，建立相应功能来记录用户已经找到了哪些花格布。

然后再回来完成这一点，让应用提示用户拍照。



我怎么让应用“记住”用户声称看到的花格布呢？我们是不是得动手写原生代码了？

很幸运，在JavaScript中可以使用一个HTML5 Web标准（名为localStorage）做到这一点。

PhoneGap Build支持的所有平台的默认移动浏览器中都支持这个标准。

在BlackBerry上OS 6之前还不支持localStrorage，不过我们目前不打算支持BlackBerry。

看到了什么？存储找到的花格布

我们可以在客户端存储简单的数据（在这里就是存储已经找到的花格布），而不用那么麻烦去使用浏览器的localStorage API。

是什么让localStorage这么特殊？

过去，开发人员经常依赖于HTTP cookie来存储需要在客户端保留的数据。不过，使用cookie有一些缺点。

每次客户对服务器做出请求时，这个域所有cookie的全部内容都会传输。有时，开发人员可能想在客户端上存储大量数据，比如图像或大量配置信息，这些用cookie存储是不可行的（或者，至少性能让人无法接受）。

另外，使用JavaScript处理cookie很遭诟病，处理很麻烦，也很笨拙。另外，我们希望放在客户端上的一些数据可能根本不需要服务器知道（或者，也许就像这里一样，根本没有服务器）。

localStorage专门设计在客户端以键-值对的形式直接存储和获取字符串数据。它为我们提供了一些方法来设置、获取和清除数据，仅此而已。一点都不复杂。

我们的应用现在有一个启动画面了！这是在一个iPod Touch上显示的启动画面。





下面是config.xml文件的最后部分，这里增加了启动画面。通过提供两个不同大小的图像，这样在分辨率较高的设备上可以显示较大的版本。

```
  <icon src="images/icon-48.png" height="48" width="48" />
  <icon src="images/icon-36.png" height="36" width="36" />

  <gap:splash src="images/splash2x.png" width="640" height="960" />
  <gap:splash src="images/splash.png" width="320" height="480" />
</widget>
```

config.xml

localStorage能为我们做什么？

参赛者单击“I found it!”按钮时，我们可以为localStorage增加一项。如果我们想看看参赛者已经找到了哪些花格布，可以查看localStorage中的数据。

认识获取方法和设置方法

localStorage的功能主要由它的两个方法提供。首先，我们可以设置一个值：

键和值都必须是字符串。

用一个键对要存储的信息命名。

我们想要存储的值。

```
localStorage.setItem(key, value);
```

然后，我们可以使用键来访问这个值：

请告诉我对应所存储数据的键！

```
var storedValue = localStorage.getItem(key);
```

如果对应这个key找到一个值，将赋给storedValue。

如果没有找到对应key的数据，storedValue将为null。

在这里，比如说用户在一个供应商的摊位找到了Douglas花格布，并轻击“I found it!”按钮指示他找到了，我们可以这样做：

```
localStorage.setItem('douglas', 'true');
```

然后如果我们想查看他是否已经找到Douglas花格布，可以这样询问localStorage：

```
var isFound = localStorage.getItem('douglas');
```



localStorage JavaScript磁贴

下面更新scripts/app.js（这个应用的主JS代码），记录找到的花格布。更新的app.js文件在下一页上。你的任务是在相应的代码行上面增加注释磁贴。

每个磁贴只能使用一次，不过最后可能有一些磁贴根本用不上！

// Get the entry from localStorage

// Click handler for "I found it" button

// Clear all entries from localStorage

// Turn off jQM page transitions

// Create a button-styled <a> element

// Add a back button to the nested list subpages

// Check for localStorage support in the browser

// Insert the reset button into the page

// Store that this tartan was found

// Add a click handler for the "I found it!" buttons

// Call the initDevice function when the DOM is ready

// Get the ID of the clicked button

// Add a click handler for the reset button

// Update the display to show which tartans have been found

```
(function() {
  $(document).bind("mobileinit", function() {

    $.extend($.mobile, { defaultPageTransition: 'none' });

    $.mobile.page.prototype.options.addBackBtn = true;
  });

  var initDevice = function() {

    if (typeof(window.localStorage) == 'object') {

      $('.foundTartan').click(tartanFound);

      addResetButton();
    }
  };

  $(document).ready(initDevice);

  var tartanFound = function(event) {

    var tartanKey = $(event.currentTarget).attr('id');

    localStorage.setItem(tartanKey, 'true');
  };


  var addResetButton = function() {

    var $resetButton = $('<a></a>').attr('data-role','button').html('start Over!');

    $resetButton.click(function() {

      localStorage.clear();

    });

    $resetButton.appendTo($('#booths'));
  };
})();
```



app.js

localStorage JavaScript磁贴答案

```
(function() {
  $(document).bind("mobileinit", function() {
    // Turn off jQM page transitions
    $.extend($.mobile, { defaultPageTransition: 'none' });
    // Add a back button to the nested list subpages
    $.mobile.page.prototype.options.addBackBtn = true;
  });

  var initDevice = function() {
    // Check for localStorage support in the browser
    if(typeof(window.localStorage) == 'object') {
      // Add a click handler for the "I found it!" buttons
      $('.foundTartan').click(tartanFound);

      addResetButton();
    }
  };
  // Call the initDevice function when the DOM is ready
  $(document).ready(initDevice);
  // Click handler for "I found it" button
  var tartanFound = function(event) {
    // Get the ID of the clicked button
    var tartanKey = $(event.currentTarget).attr('id');
    // Store that this tartan was found
    localStorage.setItem(tartanKey, 'true');
  };

  var addResetButton = function() {
    // Create a button-styled <a> element
    var $resetButton = $('<a></a>').attr('data-role','button').html('start Over!');
    // Add a click handler for the reset button
    $resetButton.click(function() {
      // Clear all entries from localStorage
      localStorage.clear();
    });
    // Insert the reset button into the page
    $resetButton.appendTo($('#booths'));
  };
})();
```

我们关闭了页面过渡效果，因为有些Android设备上页面过渡会很慢。

我们只为支持localStorage的浏览器增加click处理程序。

reset按钮也是一样。

值'true'有点随意。我们只是想存储"一点东西"。

可以看到，这里增加了一个按钮，允许用户重置，重新开始游戏。

嘿！我们还没有介绍clear()方法，你能自己想明白吗？

顾名思义，它的作用就是清除localStorage中所有的键及其关联的值。

app.js

检查浏览器支持什么

我们在第2章谈到了一些客户端特性检测，这里在initDevice中我们还要做这个工作。通过检查window.localStorage是否是一个对象，就是在检测浏览器是否支持localStorage特性。

```
var initDevice = function() {
      if (typeof(window.localStorage) == 'object') {
            $('.foundTartan').click(tartanFound);
            addResetButton();
      }
}
```

这里我们完成一些客户端特性检测，在增加click处理程序和显示reset按钮之前先确保支持localStorage。

initDevice在$(document).ready().Translation上调用，jQuery完成页面DOM初始化时就会执行这个函数。

客户端特性检测可能很简单，就像这个例子一样，不过还有一些JavaScript库可以为所有特性提供检测。Modernizr (http://modernizr.com)就是这样一个得到广泛使用的工具。

我们还在第5章利用WURFL设备能力数据来完成特性检测。那是服务器端特性检测。

不过请等等……故事还没完呢

剩下没有用到的注释磁贴给我们一个提示，从中可以看出这里还需要做些什么。我们要存储找到的花格布，另外要提供一种方法将它们全部清除，不过界面不变。我们要编写一些代码更新显示，使参赛者能够看到他们已经找到了哪些花格布。翻开下一页开始吧。

剩下的磁贴。我们需要完成这些工作。

```
// Get the entry from localStorage
```

```
// Update the display to
// show which tartans
// have been found
```

BRAIN POWER

只有在支持localStorage的情况下才增加click处理程序和reset按钮，这样实际上我们为支持localStorage的浏览器设置了一个“皮肤”，这是第4章的风格。你能想到为什么可以这样？另外为什么有些情况下不可以？

用一个函数显示哪些花格布已经找到

看来我们需要用JavaScript另外编写一个函数，根据已经找到的花格布更新页面的外观。下面来深入研究这个函数。我们要把这个新函数命名为refreshTartans，因为它会根据已经找到的花格布更新花格布列表外观和详细信息屏幕。

index.html中的各个嵌套列表（ul.details）分别包含一个供应商的有关信息和要寻找的花格布。我们用各个列表的id属性来确定一个key，用来在localStorage中查找信息。如果对应这个key有某个值，说明这个花格布已经找到，我们要更新界面反映这一点。

成品 JavaScript

那么，对于文档中的各个ul.details，它会……

这是jQuery代码。它表示迭代（循环）处理与CSS选择器ul.details匹配的各个元素（类为“details”的<ul>）。

app.js

```
$(document).ready(initDevice);
var refreshTartans = function() {
  $('ul.details').each(function() {
    var myID         = $(this).attr('id');
    var tartanKey    = 'found-' + myID;
    var foundValue   = localStorage.getItem(tartanKey);
    var isFound      = Boolean (foundValue);
    $('#vendor-'+ myID).toggleClass('found', isFound);
    $('[data-url*="'+myID+'"]').toggleClass('found',isFound);
    $('#'+tartanKey).closest('li').toggle(!isFound);
  });
  $('ul').each(function() {
    if ($(this).data('listview')) {
      $(this).listview('refresh');
    }
  });
};
var tartanFound = function (event) {
```

1. 得到这个<ul>的id属性，并在前面加上前缀'found-'，确定要在localStorage中查找的键名。

   例如'found-douglas'

2. 在localStorage中查找这个键。

3. 切换一些元素的可见性和类，来反映它们是否已经找到。

4. 刷新文档中的jQuery Mobile listview。

如果没有刷新，修改的内容就不会有正确的样式。

要记住，jQuery Mobile建立在jQuery基础之上，所以jQuery的所有方法都可以使用。

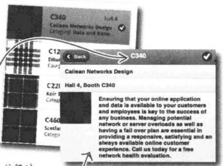


这些切换是什么？

toggle和toggleClass是jQuery的方法。下面来仔细分析。

toggle和toggleClass方法

toggle和toggleClass是jQuery方法。toggle会切换一个元素的可见性；toggleClass则切换对一个元素应用的CSS类。



通过使用toggleClass有选择地对某些元素应用"found"CSS类，这样可以利用样式来显示哪些花格布已经找到……

……另外不再需要"I found it!"按钮时，可以使用toggle在将其隐藏。

（这里没有按钮！）

下面再来看这个代码块

应该记得，localStorage.getItem(tartanKey)可能返回对应这个键存储的值（在这里就是字符串'true'），或者返回null。我们将它转换为一个Boolean值（true或false），这样就能对jQuery的toggle和toggleClass方法使用这个值了。

要使用这些方法，我们需要一个真正的Boolean值（而不是字符串'true'或null）。

如果localStorage中存在对应这个花格布的值，isFound返回true，否则返回false。

```
var foundValue   = localStorage.getItem(tartanKey);
var isFound      = Boolean (foundValue);
$('#vendor-'+ myID).toggleClass('found', isFound);
$('[data-url*="'+myID+'"]').toggleClass('found',isFound);
$('#'+tartanKey).closest('li').toggle(!isFound);
```

如果isFound值为true，toggleClass会为与给定选择器匹配的元素增加指定的CSS类(found)，如果为false，则删除这个类。

类似地，如果传入一个Boolean true值，toggle会显示指定的<li>元素。这里我们很聪明，用了一个小技巧，给定isFound当前值的相反值（这正是!isFound的作用）。你可能会问为什么？嗯，我们希望如果这个花格布已经找到（即isFound为true），就将相应的<li>隐藏，也就是包含"I found it!"按钮的那个<li>。既然这个花格布已经找到，就没有必要再显示这个按钮了。

总结：如果找到这个花格布，为这两个元素增加found类，并隐藏包含"I found it!"按钮的<li>。如果没有找到，就删除这个类，并显示这个按钮。



在scripts/app.js中增加完整的磁贴代码和refreshTartans函数。refreshTartans需要在页面初始化时调用，或者只要localStorage有修改就要调用这个函数。看看你能不能找出这3个refreshTartans调用要分别放在代码的哪个位置。

Exercise Solution

要在app.js的以下3个位置调用refreshTartans。

初始化时……

```
var initDevice = function() {
  if (typeof(window.localStorage) == 'object') {
    $('.foundTartan').click(tartanFound);
    refreshTartans();
    addResetButton();
  }
};
```

只要找到一个新的花格布……

```
var tartanFound = function(event) {
  var tartanKey = $(event.currentTarget).attr('id');
  localStorage.setItem(tartanKey, 'true');
  refreshTartans();
};
```

……以及花格布重置时。

```
var addResetButton = function() {
  var $resetButton = $('<a></a>').attr('data-role','button').
html('start Over!');
  $resetButton.click(function() {
    localStorage.clear();
    refreshTartans();
  });
  $resetButton.appendTo($('#booths'));
};
```

app.js

试一试

编辑app.js，加入前面几页的修改。保存文件，并在一个Web浏览器中预览Tartan Hunt的index.html（应该能很好地工作）。试着点击某些花格布和相应的“I found it!”按钮。应该能看到找到的花格布及其详细页面的某些CSS样式会发生改变（它们会变绿）。

如果出现问题，可以使用Chrome或Safari中的Web Inspector工具，或者使用Firefox中的Error Console，检查可能的JavaScript错误。如果确实有困难，可以在chapter8/extras/scripts/app.localStorage.js找到这个文件的一个完成版本。

如果想使用这个文件，你要把你的app.js替换为这个文件。

动手做！

再来压缩chapter8目录的内容，在PhoneGap Build上重新构建应用。然后再在一个设备或模拟器上安装，试试看！

there are no Dumb Questions

问：目前哪些浏览器支持localStorage?

答： 简单地说：大多数浏览器都支持。不过不包括Opera Mini，另外如果你还在使用Internet Explorer 7，它也不支持localStorage。

问：这些键必须有特定的名字吗?

答： 键以及为键指定的值都必须是字符串。不过除了这个要求之外，就没有其他限制了。你可以把它们叫做你喜欢的任何名字。

问：我能存储多少数据?

答： W3C localStorage规范在这方面有些含糊。这里引用其中的说法："每个源建议最多存储5MB数据。欢迎大家的实现反馈，以便将来更新调整这个建议。"

大多数浏览器都提供2到5MB。对于某些浏览器，如Safari，如果分配的存储空间用光，会提示用户分配更多空间。

问：其他网站或应用能访问我的localStorage数据吗?

答： 不行。规范中有一部分有关安全性，对某些方面做了强制要求，不允许其他源（可以粗略地解释为其他网站）除他们自己的数据外访问任何其他localStorage数据。

问：你说你只能在localStorage中存储字符串。不过，之前你提到过，可以使用localStorage存储图像。这是怎么做到的?

答： 字符串，没错。不过我们完全可以存储非常大的字符串！图像可以存储为其BASE64编码的字符串，直接用作为<img> src属性的值，或者作为CSS背景图像中的url()值。浏览器对数据URI（这里没有写错）提供了很好的支持，只有Internet Explorer是个突出的例外。有关的更多内容可以看看Nicholas Zakas的这篇文章：http://bit.ly/sWe7HS。

问：等一下！我刚刚又看了代码，注意到我们在嵌套列表页面上增加了一个后退按钮。对于Android设备这是没有意义的，大多数Android设备都已经有一个硬件后退按钮了。

答： 你的目光真犀利。在Android设备上，后退按钮不仅看着很丑陋，实际上它还会关闭PhoneGap Build生成的应用！这可不好。稍后我们还会再来讨论和修正这个问题。

问：如果浏览器中可以用localStorage，为什么我们还要用PhoneGap Build构建应用呢？不能直接让它作为一个Web应用吗?

答： 哈！耐心点！我们就要谈到这个问题了。现在该在Tartan Hunt加入相机了。

你找到了一个花格布，是吗？那就证明看看！

- [x] 建立并配置一个PhoneGap Build项目。打包压缩当前的HTML、CSS和图像；构建应用，并在Android设备或模拟器上安装这个应用。
- [x] 增加相应功能，允许参赛者标记他们已经找到的花格布。 ← 我们利用localStorage做到了这一步！
- [ ] **增加相应功能，允许参赛者保存他们找到的花格布的照片。** ← 现在怎么完成这一步呢？

有一个这方面的API

可以利用W3C mediaCapture API访问设备的音频、视频和相机记录能力。它通过JavaScript中的navigator.device和navigator.device.capture对象提供这种访问。

请等等。既然可以通过mediaCapture API在浏览器中访问相机，为什么还要实现原生应用呢？



确实有这样一个规范，不过它还很不成熟。

目前还没有哪个平台真正支持mediaCapture API。如果我们想用JavaScript访问相机，还是需要一些原生支持。

释放PhoneGap的能量

到目前为止，我们还没有真正使用PhoneGap API！PhoneGap Build已经做了一些工作，它能以原生方式封装和打包我们的Web代码。不过我们还没有利用PhoneGap来访问设备特性。该有点变化了！

在PhoneGapBuild生成的原生应用包中，可以找到phonegap.js，利用这个跨平台的API可以访问那些吸引人的特性。要使用这个API，只需要在你的HTML中包含这个脚本。

利用PhoneGap拍照

PhoneGap的JavaScript API中有很多内容，包括很多重要的JavaScript事件。我们真正关心的事件是deviceready，PhoneGap认为设备一切准备就绪时就会触发这个事件。可以在我们自己的app.js代码中查找这个事件。

我们希望把deviceready事件绑定到一个回调函数，在这个函数中具体访问相机。

需要包含phonegap.js，并监听deviceready事件。好戏就要开始了。

通过监听deviceready事件，可以确保设备已经准备好！

这个参数深层次的含义相当复杂，不过基本说来就是忽略这个事件常规的"捕获"流，而是当它在DOM层次体系上"出现"时将其捕获。

```
document.addEventListener('deviceready', initPhoneGap, false);
```

不明白也没关系。

这里表示我们希望在deviceready事件触发时执行initPhoneGap函数。

嘿，我们需要写这个initPhoneGap函数。接下来就要写这个函数了！



Exercise

试着做好准备，使用PhoneGap。

1. **更新index.html，增加phonegap.js。**

   你会注意到，chapter8文件夹中并没有phonegap.js文件。PhoneGap Build在生成原生应用时，会负责为各个原生应用增加这个脚本的正确版本。增加<script>标记，一旦我们构建应用，这个脚本就能很好地工作。

2. **在app.js中的initDevice上面创建一个名为initPhoneGap的空函数。**

   稍后将填写这个函数。

3. **增加（上面的）deviceready事件监听代码段，把它增加到initDevice函数的最后。**

   这样一来，PhoneGap初始化设备时就会调用initPhoneGap。

Exercise Solution

现在来看看更新后的app.js，另外还在index.html文件中包含了phonegap.js。还有几件事要做，然后就能真正访问相机了。

```
(function() {
  $(document).bind("mobileinit", function() {
    $.extend($.mobile, { defaultPageTransition: 'none' });
    $.mobile.page.prototype.options.addBackBtn = true;
  });
  var initPhoneGap = function() {
  };
  var initDevice = function() {
    if (typeof(window.localStorage) == 'object') {
      $('.foundTartan').click(tartanFound);
      refreshTartans();
      addResetButton();
    }
    document.addEventListener('deviceready', initPhoneGap, false);
  };
  $(document).ready(initDevice);
```

app.js

```
<link rel="stylesheet" href="style/jquerymobile1_0_min.css" />
<link rel="stylesheet" href="style/app.css" />
<script src="phonegap.js"></script>
<script src="scripts/jquery1_7_min.js"></script>
<script src="scripts/app.js"></script>
<script src="scripts/jquerymobile1_0_min.js"></script>
```

index.html

可能有点糊涂，因为我们自己的代码基中没有phonegap.js，不过一旦应用构建，这个脚本就有了。这确实……能正常工作。

噢，看呐，我们上次查看之后jQuery Mobile已经发布了1.0版本。

PhoneGap基本准备就绪，可以出场了

在app.js中再增加一些新技巧。下面该填写那个initPhoneGap函数了，是吗？

这里设置这个变量，以便JS的其他部分访问。

我们会用这个变量跟踪记录是否支持图像捕获。初始值为false。

app.js

```
(function() {
  var imageCaptureSupported = false;
  $(document).bind("mobileinit", function() {
    $.extend($.mobile, { defaultPageTransition: 'none' });
    $.mobile.page.prototype.options.addBackBtn = true;
  });

  var initPhoneGap = function() {
    if (!navigator.device || !navigator.device.capture) { return; }
    imageCaptureSupported = true;
    if (device.platform && device.platform == 'Android') {
      $('body').addClass('android');
    }
  };
  var initDevice = function() {
```

如果我们需要的某些特性不可用，就不能继续执行。

OK，如果通过这些测试，说明提供了mediaCapture支持。

太棒了！可以检查PhoneGap的device.platform属性，对Android设备隐藏后退按钮！

我们的style/app.css中有一个规则，<body>元素的类为Android时会隐藏后退按钮。

在initPhoneGap函数（也就是deviceready回调）中，我们通过两重检查确认是否能访问navigator.device.capture，并将imageCaptureSupported设置为true来指示支持这个图像捕获特性。

然后（这一点太棒了），我们可以在Android平台上隐藏不必要的后退按钮（而且实际上这个按钮会有问题）。大多数Android设备都有硬件后退按钮，重复提供后退按钮会让用户困惑。不过如果希望在iOS设备上使用这个应用，就确实需要这样一个后退按钮。否则，用户没有办法回到前一个屏幕。这个小代码块可以解决我们的问题。

动手做！

好的，可以开始了！把这些准备工作放在app.js中，现在准备，灯光、摄像、开拍！

现在准备使用mediaCapture API

与很多PhoneGap设备API一样，PhoneGap mediaCapture API模拟了W3C规范的mediaCapture支持。这个规范中有很多内容，其中有一个方法能帮助我们完成工作。下面就来认识认识这个imageCapture方法。

成功时要调用的函数！

选项！例如，如果希望相机提示用户只拍一张照片，这样我们会把它设置为一个选项。

```
navigator.device.capture.captureImage(successCallback, errorCallback, options);
```

captureImage启动设备的相机，并拍照。

出现问题时要调用的函数。

很好，不过这个代码放在哪里？这些成功和错误函数里需要些什么？这里我还需要一些帮助。



OK。下面来看我们想要达到什么目的。我们希望参赛者单击“I found it!”按钮时触发这个事件。

它应该在设备上打开原生相机应用，让用户拍一张照片。一旦参赛者对这张照片满意，表示确认，就回到我们的Tartan Hunt应用，这里会发生两件事：将这个花格布记录为“已找到”，另外刚拍摄的照片会出现在详细页面上。

要做的第一件事是更新tartanFound函数，这是“I found it!”按钮click事件的处理程序，它使用captureImage从用户的相机得到一张照片。就从这里开始吧！

成功时如何处理？

参赛者点击“I found it!”按钮时，假设他有一个带相机的设备，而且PhoneGap Build mediaCapture提供支持，我们希望调用captureImage。

应该记得，captureImage的第一个参数是一个函数，如果一切正常会调用这个函数。理论上讲，我们的成功回调函数应该是这样：

嗯……这个mediaFiles是什么？

mediaFiles[0].fullPath会为我们提供mediaFiles数组中第一个图像的路径。

```
function itWorked(mediaFiles) {
  localStorage.setItem(tartanKey, mediaFiles[0].fullPath);
}
```

因为我们设置了一个选项，要求只拍一张照片（有关内容稍后介绍），所以这个数组中只有一项。

captureImage成功时，它会调用已定义的成功回调，为这个回调函数传入一个MediaFile对象列表（JavaScript数组）。MediaFile并不包含文件，实际上，它包含了大量相关的元数据。不要灰心，因为其中已经包含我们需要的东西：图像文件在设备文件系统中的路径。另外这里不再保存没意思的true值，我们可以在localStorage中保存这个图像的路径以备以后使用。

如果出了问题

如果相机使用不正常，比如说用户取消了相机功能，或者出了其他问题，我们还需要有一个错误回调函数。由于拍照时出问题不会造成太严重的后果，所以我们力求简单，只是把它记录到控制台上，这样感兴趣的人可以进一步调试。所以，错误回调可以像这样：

这只是一个错误回调的例子。

```
function ohSad(error) {
  console.log(error);
}
```

真实世界里总有点不同

准备认识扩展的tartanFound函数？要记住，参赛者单击某个花格布相应的“I found it!”按钮时就会执行这个函数。嗯，下面给出这个函数。必须承认，这里的更新看起来与之前展示的例子稍有些不同。

我们在initPhoneGap函数中设置imageCaptureSupported，还记得吗？

如果支持mediaCapture，而且这个特性可用，可以拍照了。

对于不支持这个特性的设备，只使用之前一直用到的“true”值标志花格布“已找到”。

```
var tartanFound = function(event) {
  var tartanKey = $(event.currentTarget).attr('id');
  if(imageCaptureSupported) {
    navigator.device.capture.captureImage(function(mediaFiles) {
      localStorage.setItem(tartanKey, mediaFiles[0].fullPath);
      refreshTartans();
    }, captureError, {limit:1});
  }
  else {
    localStorage.setItem(tartanKey, 'true');
    refreshTartans();
  }
};
var captureError = function(error) { console.log(error);  }
addResetButton = function() {
```

在这里设置选项，来限制照片数量为1。

我们的错误回调。

app.js



因为我们想要使用结合上下文的tartanKey当前值，所以这里使用了一个匿名函数作为我们的成功回调。

只是匿名

这里不再为captureImage指定成功回调函数名，我们会直接在那里定义整个函数。

你可能会问，这有什么意义？看看我们如何把所点击的元素（event.currentTarget，也就是“I found it!”按钮）的id属性赋给tartanKey变量？通过把成功回调函数定义为一个匿名函数，我们可以根据单击了哪个按钮访问相应的tartanKey值。如果使用tartanFound函数之外定义的一个函数，就不能在当前上下文中访问这个值。

tartanKey是所点击的按钮的id属性。

这里突出显示的代码是成功回调函数。

```
var tartanKey = $(event.currentTarget).attr('id');
if(imageCaptureSupported) {
    navigator.device.capture.captureImage(function(mediaFiles) {
      localStorage.setItem(tartanKey, mediaFiles[0].fullPath);
      refreshTartans();
    }, captureError, {limit:1});
  }
```

我们指定了错误回调函数名（这个函数在别处定义，见第350页）。

这里引用了tartanKey，可以从所单击的按钮得到tartanKey (id)。

把刚从相机生成的图像的完整路径存储在localStorage中。

BRAIN POWER

在我们的代码中，imageCapture成功函数中更新了localStorage。如果出于某种原因没能得到照片，花格布不会标志为已找到。

你认为这样对吗？有没有更好的解决方法？



OK，我知道要把图像的路径存储在localStorage中。不过怎么使用呢？

很高兴你能问这个问题。我们需要更新refreshTartans函数。

如果用户为一个给定的花格布拍了一张照片，要把这张照片显示在相应供应商摊位的详细页面上。

重要的是，可以用我们存储的路径作为img元素的src属性。这并不难。

localStorage中有一个照片路径时，可以创建一个jQuery <img>元素，把我们存储的路径用作为这个元素的src属性，将它放在页面的一个新列表元素中。

成品 JavaScript

```
var refreshTartans = function() {
  $('ul.details').each(function() {
    var myID          = $(this).attr('id');
    var tartanKey     = 'found-' + myID;
    var foundValue    = localStorage.getItem(tartanKey);
    var isFound       = Boolean (foundValue);
    $('#vendor-'+ myID).toggleClass('found', isFound);
    $('[data-url*="'+ myID +'"]').toggleClass('found', isFound);
    $('#'+tartanKey).closest('li').toggle(!isFound);
    var hasPhoto      = (isFound && foundValue != 'true');
    if (hasPhoto) {
      if (!$(this).find('.tartanImage').length) {
        var $tartanHolder = $('<p></p>').append($('<img>').attr({
          'src'    : foundValue,
          'class'  : 'tartanImage'
        }));
        $(this).append('<li data-role="list-divider">My Photo of the Tartan!</li>');
        $('<li></li>').append($tartanHolder).appendTo($(this));
      }
    }
  });
  $('ul').each(function() {
    if ($(this).data('listview')) { $(this).listview('refresh'); }
  });
};
```

如果对应这个花格布存储了一个值，而且不是字符串 'true'，那么这必然是一个照片。

使用存储的路径作为img的src属性。



试一试

OK，准备好了吗？JavaScript中做了一些加工处理，不过最后的app.js只有大约75行代码。我想对于一个完整的应用来说，这确实相当简练！

最后一点！

给你一个小小的提示：你可能希望在config.xml文件中增加下面这两行代码。方向（orientation）首选项会保证在你启动相机应用时，设备（尤其是模拟器）不会进入水平模式。第二行代码则指示PhoneGap我们想要使用API的相机特性；它会映射到正确的Android许可设置。

```
<preference name="orientation" value="portrait" />
<feature name="http://api.phonegap.com/1.0/camera" />
```

config.xml

大功告成了。在看过前面几页大段的代码之后，深呼吸，放松一下。确保已经把所有修改加入到app.js中，并保存了对config.xml的修改。一切都准备好时，可以再次压缩chapter8文件夹，在PhoneGap Build中构建应用，然后上传更新后的代码。

构建过程一旦完成，下载APK文件，把它安装在你的模拟器或设备上。

不要忘记先卸载原来的应用！



如果在一个模拟器上安装这个应用，即使相机有点问题，也不会有大麻烦。

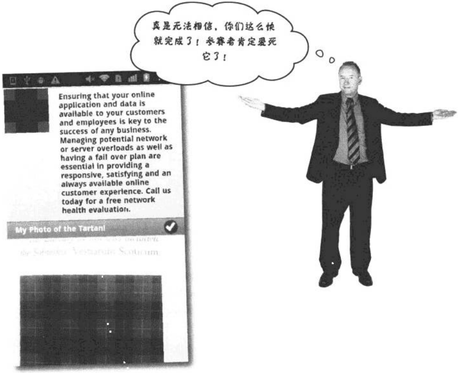
我们发现，对于Android 2.3之前版本的模拟器，相机很不稳定。有时它们会锁住或崩溃。建议你的AVD（Android虚拟设备）使用更新的Android版本。

即使在较新的模拟器上，相机体验也不完全是 惊喜。有些模拟器只会显示一个白屏。另外一些模拟器可能会给出一个粗糙的动画。不过通常情况下它们确实能成功地生成某种图像。

如果你的朋友有一个Android设备，现在可以请他帮忙。在真正的设备上相机会有更好的表现。

真的很难吗？

可以在extras/scripts/app.final.js找到app.js的最后版本。Android应用APK文件的最后版本可以在http://hf-mw.com/ch8/TartanHunt.apk找到。



搞定了！

- ☑ 建立并配置一个PhoneGap Build项目。打包压缩当前的HTML、CSS和图像；构建应用，并在Android设备或模拟器上安装这个应用。
- ☑ 增加相应功能，允许参赛者标记他们已经找到的花格布。
- ☑ 增加相应功能，允许参赛者保存他们找到的花格布的照片。



BULLET POINTS

- 移动Web的发展已经为时不短。不过还有一些特性无法在浏览器中访问。
- 混合型应用方法是指从利用HTML5、CSS和JavaScript标准编写的代码创建原生应用。混合型应用可以减少原生开发开销和复杂性，因为代码可以跨平台共享。
- PhoneGap是一个用于构建原生应用的HTML5平台。它由总部在温哥华的Nitobi公司创建。Nitobi于2011年秋天被Adobe收购。
- PhoneGap Build是一个服务，允许你在一个简单的web界面中构建PhoneGap项目。
- PhoneGap Build项目的组织类似于W3C Web部件。它们至少需要一个开始文件（index.html）和一个配置文件（config.xml）。
- W3C localStorage API允许在客户端直接存储键-值字符串数据。我们用它指示参赛者已经找到哪些花格布。
- phonegap.js会自动增加到PhoneGap Build项目，使我们能够访问PhoneGap JavaScript API。我们已经了解如何从PhoneGap监听deviceready事件，从而知道设备特性已经准备好，可以发挥作用了。
- 为了对我们的应用增加相机支持，这里使用了PhoneGap的mediaCapture API，它效仿了W3C mediaCapture API。
- 我们使用navigator.device.capture.captureImage方法创建原生相机应用并拍照。
- 我们只需要区区75行JavaScript代码和一个HTML页面就创建了一个跨平台的原生应用！

9 如何实现未来友好

应对混乱



响应式Web设计、设备检测、移动Web应用、PhoneGap。等一下……我们该用哪一个呢？移动Web开发方法实在太多了。通常，项目中都会结合应用多种技术。这方面并没有惟一的正确答案。不过不用担心。关键是要学会“随波逐流”。掌握不确定性。用一种未来友好的思维，顺应潮流，确保一定要灵活，能够适应未来的一切变化。

现在怎么样呢？

我们介绍了很多内容，从网站到应用，从响应式Web设计到设备检测。要为移动设备构建应用，有很多不同的方法。你究竟如何选择？

移动Web策略

- 响应式Web设计
- 移动优先响应式Web设计
- 渐进增强
- 单独的移动网站
- 设备检测
- 设备类
- 移动Web应用
- jQuery Mobile
- 离线支持
- 超级移动Web应用
- PhoneGap/Apache Cordova


我们想使用移动优先响应式Web设计，不过我们没办法使用一个第三方服务器提供每一个图像，另外目前所有JavaScript解决方案看起来都不可靠。是不是前景很黯淡？
每天都有新设备推出。设备检测就像是一个永无止境的军备竞赛。
平板设备呢？必须等到Head First Tablet Web出版才能了解如何为iPad构建Web页面吗？
我们需要一个网站，还需要应用。单单测试费用就很高。


啊！快停下！

太复杂了

我们必须面对这一切。随着设备数量的增多，Web开发已经变得复杂得多。更糟糕的是，在情况好转之前，Web开发可能会变得越来越困难。

如今发生着日新月异的变化。不仅新设备不断引入，这些设备中的技术和软件也在飞速地变化。

甚至我们与计算机交互的方式也受到挑战，触摸式界面、相机输入和语音传感器为我们展示了新的工作途径。

我们该怎么做呢？面对这种不确定性，该如何设计让用户趋之若鹜的体验呢？

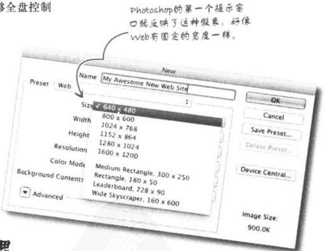
消除控制假象

长期以来，我们一直都在欺骗自己，以为我们能够全盘控制Web。

我们的工具和处理方法中都充分反映出这种假像。通常，开始开发项目时，我们可能先在Photoshop中创建一个模拟应用。Photoshop首先会询问我们要建立多大的画布。

不过，没有哪一种画布大小能够适应各种情况。甚至对于桌面应用都没有一种各种情况都适用的大小设置。

越早了解这个现实，就能越早开始学习如何构建能够适应环境和用户的网站和应用。



我们需要一种不同的思维，要能够接受多样性，着眼于适应未来可能出现的各种未知情况。

未来友好宣言

你并不孤单。一大群移动开发人员被召集起来，大家一同头脑风暴，想办法应对Web的这种不断变化。召集的结果就是形成了这个未来友好宣言。

这个宣言可以在http://futurefriend.ly找到。

未来友好



在当今这个不可思议而又充分困惑的互连数字设备的世界里，我们要清楚以下事实才能做到先知先觉：

纷乱只会更为加剧。互连设备（其中很多我们以前无从想过）的数量和种类将呈爆炸式增长，同样的，全世界使用这些设备的人无论从人数还是差异性来讲都在飞速增加。

我们现有的标准、工作流和基础设施已经无力支撑。如今，设备的巨大冲击已经把这些标准推到潮头浪尖。对于前面的未知世界它们无力应对。

专用的解决方案起初会占据统领地位。创新必须先行，才能引导标准化进程。技术专家会对这些解决方案趋之若鹜，之后才（再一次）意识到必须有一个标准化的平台才能保持稳健发展。

这个标准化过程会很慢，而且这是一个非常痛苦的历程。我们要努力争取得到（并最终认可）适当的标准。在此期间，Web还会继续受专用解决方案的趋使。

新的希望

不过还是有希望的。尽管我们不能清楚地知道未来会带来什么，但是我们可以：

1. 承认并充分把握不确定性。
2. 用一种未来友好的方式去思考、去行动。
3. 帮助别人也这样做。

只要友好，未来就是我们的。

签名



LUKE WROBLEWSKI	SCOTT JENSON
BRAD FROST	JEREMY KEITH
LYZA D. GARDNER	SCOTT JEHL
STEPHANIE RIEGER	JASON GRIGSBY
BRYAN RIEGER	JOSH CLARK

Home · Thinking · Resources

我们是 Creative Commons friendly. **欢迎联系**

如果你不能保证未来，那就实现未来友好吧

Web未来会朝着什么方向走？这有太多的不确定性，所以我们必须构建尽可能灵活的网站和应用。这说明，尽管是在当下构建应用，但要一直着眼于未来。



由于不断催生出现新的设备，所以做到这一点并不容易。面对这种复杂性，与以往相比，人们更容易选择走捷径，构建那种只是今天可用的应用（也许只能在很少的设备上工作），而把统一访问和未来的设备抛在脑后，等着将来某一天再做打算。

Web开发总是需要折衷。关键是要想办法做出适当的妥协，不要以某种方式锁定你的内容和数据，以至于将来无法访问和变换。

没有万全之策

例如，移动优先响应式Web设计中对于图像就没有一种理想的解决方案。我们之所以选择Sencha.io Src（尽管它依赖于设备检测），是因为可以保证HTML标记不受影响。现在我们可以引入Sencha.io Src，如果有更好的解决方案出现，能很快地将其替换。

类似地，如果有时需要完成浏览器还不支持的工作，可以选择类似PhoneGap的技术，根据标准得到提示，而不是直接选取可能只有很短寿命的专用解决方案。

要想在未来获得成功，关键在于语义标记和渐进增强。把重点放在这些原则上，你就能取得不错的成绩。

试一试

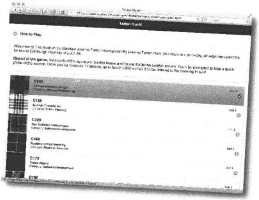
从第8章开始，Tartan Hunt中就融入了未来友好的思想，尽管这个应用构建为作为一个原生应用运行。查看Tartan Hunt代码，并在桌面浏览器中打开index.html。

这个应用能工作吗？有没有什么遗漏的东西？

如果你需要这个应用最后的代码，可以在http://hf-mw.com/ch9/chapter9.zip下载。

今天的应用，明天的网页

尽管Tartan Hunt设计作为一个原生应用，但这个应用确实可以在浏览器中运行。已经找到的花格布会记录下来，并用绿色突出显示。唯一缺少的是相机。



由于我们采用了一种未来友好的方式设计这个应用，使用的是建议的W3C标准，所以尽管应用中有些部分现在不能用，但很可能将来就能正常工作了。

删除PhoneGap引用

可以做两个很小的调整，这会让应用将来在浏览器中工作得更好。

1. **删除phonegap.js的引用。**

   等到浏览器达到登峰造极的水平，原生和Web之间不再有缺口，那时我们就不再需要phonegap.js库。尽管留下phonegap.js引用也不会造成什么破坏，不过浏览器会因此得到一个"文件未找到"（404）错误。所以我们把这个引用去掉。

   从index.html删除这一行。

```
<script src="phonegap.js"></script>
```

类似这样的策略就是未来友好思想的核心……

2. **修改页面初始化函数。**

   PhoneGap、jQuery和jQuery Mobile可能在页面初始化和JavaScript事件创建的顺序方面出现不一致。简单地说，不能指望它们肯定按某个特定的顺序触发，所以Tartan Hunt中专门编写了一些代码来强制顺序。

   不过不用担心，要让JavaScript保证顺序，并不需要太多的技巧。你只需要修改一行代码。

把这行代码……

```
document.addEventListener('deviceready', initPhoneGap, false);
```

……改为这样。

```
initPhoneGap();
```

随便说一句，这个函数名如果不提到PhoneGap会更好（啊）！

一旦浏览器支持mediaCapture API，Tartan Hunt就能做好准备，投入使用！

漫长的道路：这里有一些路标

是不是希望想办法把那个未来友好宣言变成真正实用的策略？下面给出了http://futurefriend.ly提供的一想法，可供参考。这些并不是规定，因为没有人知道未来将会怎样，不过这些确实是需要关注的领域，要在你的工作中重点考虑。

精准关注

我们不可能在所有设备上实现所有功能。在这样一个设备复杂性不断增加的世界里，为了有效地进行管理，我们应当关注那些对顾客和企业最重要的方面。不要只是构建满足最低需求的解决方案，而应当创建有意义的内容和服务。人们会越来越厌倦过多的额外内容（对他们来说，这些可能只是噪声），并想办法进行简化。不断增加的多样性可能也要求精简你的服务，在此之前，以及在顾客要求之前，应当明确你的服务的重点。

交换数据

设备系统要求是可互操作的，而健壮的数据交换是确保互操作性最容易的方法。通过以下方式定义你的数据，从而可以对现有以及将来可能出现的设备做出响应：

- 支持多种（灵活）形式的访问和通知。
- 使用标准，做到可互操作。
- 强调长期完整性。
- 包含所有内容的（有意义的）持久引用。
- 支持读写操作。

通用内容

良构的内容现在已经成为版面设计的一个重要部分。要考虑内容如何“流入”各种容器，为此要注意它们的约束和能力。要大胆一些，努力研究新的可能性，不过还要清楚未来可能会朝着多个方向发展。

未来我们要面对功能极其强大的智能设备、简单的受限设备、可以互操作的设备，以及（很多）其他设备，这些都是未来的一部分。所以要相应地建立你的内容结构并存储内容。

未知对象

要对每一种设备做出响应，这使得全面性设计极有难度。对应不同设备类型提供一组足够闭合的高级标准，这样可以简化适应性调整过程。另外，详细的简档信息可以对这些标准做出补充。

将设备类型归类可以管理当今的制造商，同时允许将来出现新的设备类型。

集体行动

我们的生活中有各种各样的设备，这使得我们可以在这些设备之间分配任务和信息。如果一个应用能够用于一个设备集，这个设备集中的每个设备都能以最佳的状态处理交互。这样就不再需要针对每一个设备调整服务的所有方面，允许我们充分利用各种设备能力。

这一点很突出。想象一下，你的手表成为手机上的第二个屏幕。

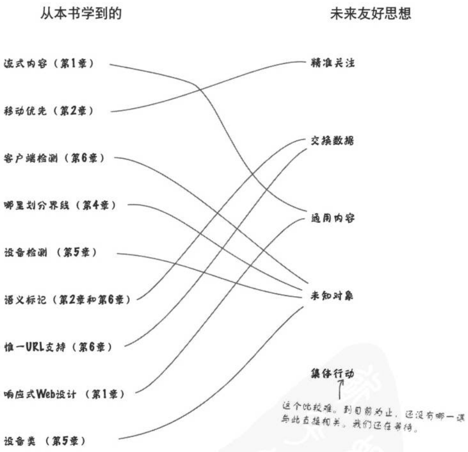
WHO DOES WHAT?

把这本书中的每一课与一个未来友好路标连线。可能每个路标对应不只一课。

从本书学到的	未来友好思想
流式内容（第1章）	精准关注
移动优先（第2章）	交换数据
客户端检测（第6章）	通用内容
哪里划分界线（第4章）	未知对象
设备检测（第5章）	集体行动
语义标记（第2章和第6章）	
唯一URL支持（第6章）	
响应式Web设计（第1章）	
设备类（第5章）	

WHO DOES WHAT? 答案

你可能认为这里有些课还用到了另外一些未来友好路标。下面是我们的答案。



Exercise

移动大餐

要制作一顿可口的移动大餐，这意味着需要结合不同配料，调制新的口味。要让这些菜肴更美味，让它们更加未来友好，应该增加哪些配料？请写在菜谱卡片上。

配料

通用内容

不要试图模仿一个原生应用。

客户端设备检测

使用hCard微格式和地址标记。

hCard是一组表示地址的标准标记。有关的更多内容可以从http://microformats.org/wiki/hcard了解。

HTML5语义元素（article、section等）。

alt属性支持可访问性的图像标记。

为用户首选项设置cookie的链接。

动物关怀网站

问题

- 需要一种方法从移动网站改进到桌面网站。
- 移动网站不包含桌面网站的全部内容。

花格布爱好者网站

问题

- 在BlackBerry 5.0上运行很慢。
- 滚动很笨拙，也很慢。
- 有两个后退按钮（一个是浏览器提供的，另一个是应用创建的）。
- 详细页面上的花格布是内容，即使用了CSS背景图像。

非凡海象网站

问题

- 所有内容都放在<div>中组织。
- 地址是由
标记分隔的文本。标记没有任何语义含义。



Exercise Solution

下面是我们的答案。你能想到还可以用哪些方法来改进这些项目，让它们更加未来友好？

动物关怀网站

问题

- 需要一种方法从移动网站改进到桌面网站。
- 移动网站不包含桌面网站的全部内容。

如果建立单独的网站，让人们选择他们想要使用哪个网站。

为用户首选项设置cookie的链接。

通用内容。

即使对于单独的网站，人们也可能想做他们在桌面网站上所能完成的一切。

没错，BlackBerry 5.0支持所需的JavaScript特性，只是很慢。检测BlackBerry 5.0并指定没有jQuery Mobile的版本会得到更好的体验。

花格布爱好者网站

问题

- 在BlackBerry 5.0上运行很慢。
- 滚动很笨拙，也很慢。
- 有两个后退按钮（一个是浏览器提供的，另一个是应用创建的）。
- 详细页面上的花格布是内容，即使用了CSS背景图像。

客户端设备检测。

不要试图模拟原生应用。

alt属性支持可访问性的图像标记。

从CSS图像移到img标记。

重复的后退按钮，还有速度很慢的滚动，这是因为试图在浏览器中模仿原生应用。如果应用不在App Store中，为什么要模仿它呢？

如果使用HTML5元素，我们的标记会包含更多含义。

非凡海象网站

问题

- 所有内容都放在<div>中组织。
- 地址是由
标记分隔的文本。标记没有任何语义含义。

HTML5语义元素（article、section等）。

使用hCard微格式和address标记。

我们只是简单提到过微格式。它们会增加语义含义。有些浏览器和搜索引擎会识别页面中的微格式，并使用这些信息提供额外的特性。



面向未来

不要纠结于将来的设备和浏览器会提供什么。我们的世界中已经有各种各样的设备。从小屏幕到大屏幕，从移动手机到支持Internet的冰箱，你要做好准备处理任何设备。

在Web标准坚实的基础上，利用渐进增强，你可以构建能让更多人访问的网站和应用。

要把握住Web可能带来的不确定性。正是由于这种不确定性，为移动带来了大量机会，同时也充满了乐趣。

继续前进吧，移动起来。建设美好的未来！





i 其他

6大主题（我们没有谈到的）



是不是感觉少了点什么？我们知道你是什么意思……你以为已经完成了，但实际上还有一些事情没有做。如果不再做一点补充（这些内容不适合放在前面的正文里），我们实在不忍心就此放手。至少如果你还没有一个带轮子的金属旅行箱，就想带着这本书闯荡世界，那是肯定不行的。来看看（还）少哪些内容。

#1 在移动设备上测试

移动测试很烦人。没错，我们只能这么说。真是没办法违心地夸它。如果一个手机在不同的承载网上有不同的表现，你就会知道情况很麻烦，你不能做任何假设。

我们会采用以下做法来尽量保持冷静。

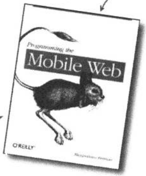
1. **在桌面浏览器中从有效的代码开始着手。**

   如果代码在桌面浏览器中出问题了，在移动浏览器上肯定也会出问题。最好先从我们更熟悉的环境开始，这些工具也更完善。一定要验证你的代码，尽早发现错误。

2. **使用移动模拟器和仿真器。**

   几乎每个移动平台都提供了一个模拟器或仿真器。Maximiliano Firtman 在他的《Programming the Mobile Web》书中整理了模拟器和仿真器的一个超强列表。
   他会在http://www.mobilexweb.com/emulators定期更新这个列表。

Max的书中包含一个超强的移动模拟器和仿真器的引用。



3. **购入少量的设备。**

   购买一些手机是不可避免的。应该由你完成的项目类型和所面向的市场来指导你确定买哪些手机。明确那些关键的投资人会使用哪些手机来检查你的工作，这些设备也需要购进。每个项目都要把购买设备的成本加在预算里。

Peter-Paul Koch写了一篇文章"A List Apart"，其中对于如何在预算吃紧的情况下选择手机给出了一些很好的建议：www.alistapart.com/articles/smartphone-browser-landscape/。

4. **讨要、借用、盗取。**

   与同样在做移动开发的人联系，共享设备。可以考虑在你的社区创建一个设备中心协会，这样人们可以很容易地找到和分享设备。更好的是，你可以做得更棒，可以建立一个社区设备测试实验室，就像我们在波兰建立的那个测试实验室一样。





5 访问当地的移动测试中心。

几乎每个城市都有一个移动设备测试中心。通常它们更常用的名字是：运营商门店。

例如，当地的Verizon手机店。

大多数运营商门店都允许人们试用设备。他们不太可能让你整天把办公室转移到他们店里，不过绝对不会阻止你在多个设备上测试你的网站。

要如实告诉他们你在做什么，并感谢他们的帮助。你会发现，销售人员还会给你提供很好的建议，告诉你哪些设备卖得好，这会对你的工作很有意义。

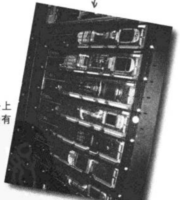
6 远程设备测试服务。

有时你确实需要测试一个特定的场景。例如，你需要一个特定的手机款式，不过这种款式只在某个特定地区才有售，仅由某个运营商独家供应。在这些情况下，唯一的办法就是利用远程设备测试服务。

Device Anywhere (http://deviceanywhere.com)和Perfecto Mobile (http://perfectomobile.com)允许远程访问世界各地数据中心的真正的手机。世界各地都有数据中心，这意味着你可以在当地的一个特定手机上测试（这个手机在当地无线提供商运营的网络上运行），并能看到使用这个手机的人在他自己的国家得到的真实体验。

这两个服务都按小时收费，这说明你要力求高效，否则会向你收取高额的测试费用。

手机已经打开，连接到加利福尼亚San Mateor Device Anywhere数据中心的服务器。你可以从世界上任何地方控制这些手机。



7 设定测试的优先级。

即使你能访问已经生产的每一个设备，也没有足够的时间在所有设备上进行测试。要选择有代表性的设备，之前在项目中已经确定了客户最有可能使用的设备，要根据为此所做的决策确定测试的优先级。

#2 远程调试

我们已经讨论过，要查看在一个移动浏览器上发生了什么，这是相当困难的。不过，即使你能看到发生了什么，要想在这样的小屏幕上完成大量应用开发也是很难想象的。

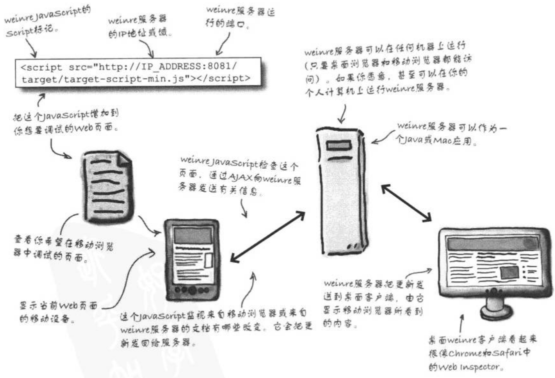
远程调试可以解决这个问题，可以把移动浏览器连接到一个桌面调试工具，这样你就能查看移动浏览器中发生了什么。

WEb INspector REmote (weinre)

远程调试有很多不同的选择，不过大多数设备和浏览器都可以使用weinre（读作winery）。weinre是一个开源项目，由WebKit资助人Patrick Mueller发起，后来被收至PhoneGap旗下。

可以在这个地址下载weinre项目：http://phonegap.github.com/weinre/。

weinre如何工作



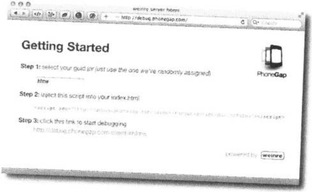
运行weinre服务器

运行weinre服务器并没有火箭科学那么高深，不过也不算太简单。幸运的是，PhoneGap有好心人已经建立了一个公共weinre服务器，可供我们使用。你只需要一个桌面浏览器和一个移动浏览器，另外要访问Web。

使用公共服务器的一个缺点是，它对所有人都是开放的。要选择一个别人不容易猜出的guid。

1 **使用Chrome或Safari访问debug.phonegap.com。**

这个页面分为3个步骤，都很容易。第1步需要提供一个guid（全局唯一标识符，globally unique identifier）。基本说来，你要选择一个词或短语，创建你自己的weinre测试服务，从而不允许别人试图使用同一个测试实例。



2 **创建一个HTML文档来完成测试。**

用以下标记创建一个新的HTML文档，保存为weinre.html。增加debug.phonegap.com提供的<script>标记，并把更新的文件放在一个公共Web服务器上。

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Weinre Example</title>
  <style type="text/css">
  h1 {color:red}
  #box {background:#ffc;padding:10px;width:70%;height:200px;}
  </style>
  <script src="http://debug.phonegap.com/target/
target-script-min.js#hfmw"></script>
</head>
<body>
<h1>Weinre test page</h1>
<p id="box">This starts as a yellow box.</p>
</body>
</html>
```



增加debug.phonegap.com提供的<script>标记。

把weinre.html放在一个公共Web服务器上，可以在这个服务器上完成测试。

出于安全原因，我们不能为你托管这个文件。



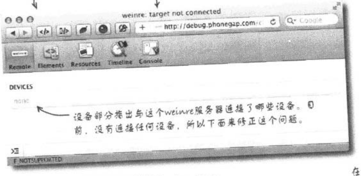
3 开始调试。

点击debug.phonegap.com上第3步下的链接，开始调试。

单击这个链接开始调试。

开始调试之后，你会看到一个类似下面的页面。

浏览器窗口标题指出没有连接目标设备。



设备部分指出与这个weinre服务器连接了哪些设备。目前，没有连接任何设备，所以下面来修正这个问题。



在一个iPhone上运行的weinre.html。

4 在一个移动浏览器中打开weinre.html。

确保浏览器支持Ajax连接。所有Android、iPhone或最新的智能手机都能正常工作。你会看到一个测试页面，其中有一个很大的黄色方框和一个红色标题。

5 验证调试器工具已经连接到设备。

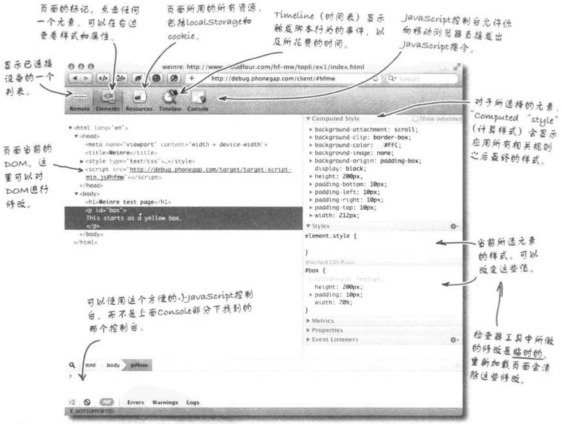
在调试器工具“远程”（remote）部分下的设备列表中，现在应该能看到你的测试设备。如果没有看到这个设备列出，可以尝试刷新调试器工具。

现在我们要测试的设备会列在这里。



研究weinre检查器工具

既然已经建立了连接，可以看到一个检查器工具，这个版本比Google Chrome和Safari提供的Web Inspector稍有些简化。





你可能以为练习已经完成了，是不是？我们也这么想，不过这是一个很有意思的练习。可以使用检查器工具对Web页面做一些修改，观察你的手机上的变化。

1. 把鼠标停在Elements（元素）标签页的HTML标记上。移动浏览器中会发生什么？

2. 使用检查器工具把背景色从黄色改为另一种颜色。

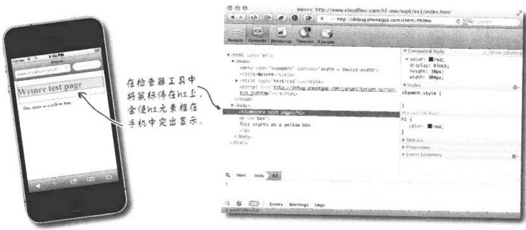
3. 使用JavaScript控制台在移动浏览器上触发一个警告。

Exercise Solution

在weinre检查器工具中做出修改后会施展一些魔法，可以观察到这些修改会立刻显示在移动浏览器上。

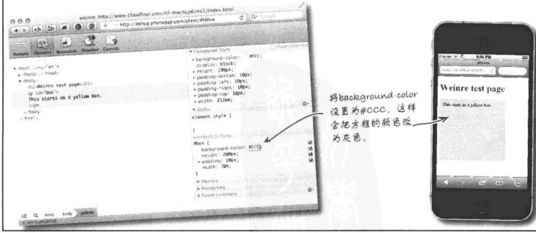
1. 把鼠标停在Elements（元素）标签页的HTML标记上。移动浏览器中会发生什么？

在桌面远程检查器工具中将鼠标停在元素上时，移动浏览器上这些元素会突出显示。

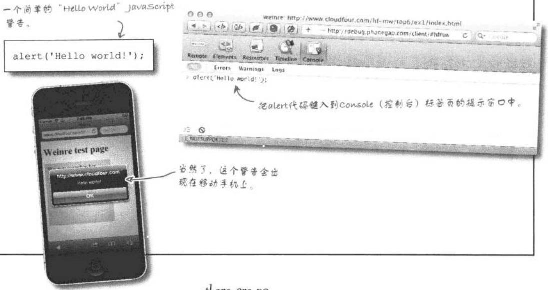


2. 使用检查器工具把背景色从黄色改为另一种颜色。

单击对应这个框的<p>标记。在相应的CSS规则部分编辑background-color属性。



3. 使用JavaScript控制台在移动浏览器上触发一个警告。



there are no Dumb Questions

问：debug.phonegap.com安全吗？为什么不托管Web页面？

答： 安全问题很少见。如果有人要使坏，他需要知道你的weinre服务器的URL，而且你那时正好在使用服务器，如果满足这些条件，他就能向你的浏览器发送恶意的JavaScript。

正像我们说的，这种情况不太可能发生，除非你公开发布了weinre服务器的URL（比如说在一本书中公开出版），并托管一个测试页面，告诉所有人去使用这个页面。发布这个页面会让它成为恶意攻击者的目标。

问：你提到了其他的调试器工具，有哪些呢？

答： Opera Dragonfly(http://opera.com/dragonfly/)有一个远程调试器工具，可以与Opera浏览器结合使用。BlackBerry Playbook和BlackBerry OS在版本7之后都对元素调试提供了内置支持。WebKit本身包含了一些工具来完成远程调试，不过大多数浏览器中都关闭了这些特性。我们写完这个附录的前一天，Maximiliano Firtman创建了iWebInspector（http://iwebinspector.com），这是一个面向iOS的远程调试器工具。我们还没有对它做过测试。

问：Adobe收购了Nitobi，也就是PhoneGap的前身，这是不是意味着 debug.phonegap.com会消失？

答： 我们已经确保这个服务不会消失。即使真有这种事情发生，你仍然能下载weinre，并在你的本地机器上运行。

#3 确定浏览器支持什么

利用渐进增强来支持能力更强的浏览器固然不错，不过如果花那么多时间构建某个功能来使用某个浏览器特性，但是只有很少数的设备能使用这个特性，这么做真的值得吗？

某些情况下，你可能希望知道哪些浏览器能够利用你基于陀螺仪构建的导航。

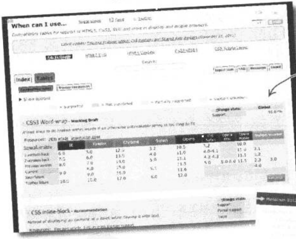
不要这么做。陀螺仪控制的导航是个很糟糕的想法！比闪烁的标记还要糟糕100倍。请你忘掉我们提到过它。

列出各个浏览器支持哪些特性

下面这些网站着力列举出各个浏览器及其支持的特性。

caniuse.com

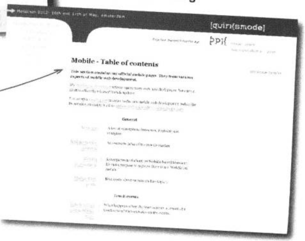
回答"我什么时候能使用……"的问题。



列出桌面和移动浏览器对于HTML、CSS和JavaScript的兼容性表格。

所有兼容性列表网站的始祖。Peter-Paul Koch (PPK)最早于1998年开始测试浏览器，并在2003年创立了QuirksMode.org。

quirksmode.org



PPK现在几乎完全关注于移动浏览器。可以读读他的博客，另外还可以查看Mobilism，这是他组织的会议。

测试你的浏览器支持什么

如果你要访问一个浏览器，想知道它支持哪些特性，下面这些网站会更有帮助。访问一次就能全面了解。

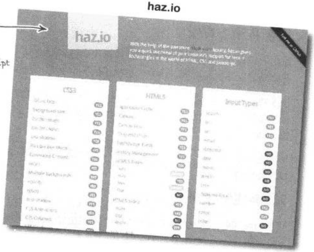
html5test.com



检查你的浏览器对一些HTML5特性的支持情况，并给出总分。

使用Modernizr库(www.modernizr.com)测试你的浏览器对HTML、CSS和JavaScript最新特性的支持情况。

haz.io



或者同时做这两件事……

browserscope.org

Browserscope会像haz.io一样测试浏览器对特性的支持，另外它还会跟踪记录哪些浏览器以前已经通过测试（或者失败）。



可以创建你自己的测试，并把数据存储在Browserscope。开发人员一直在使用Browserscope API编写更多测试，这使得Browserscope数据的价值与日俱增。

#4 设备API

尽管个头很小，但移动手机绝对是功能强大的设备，配备有很多超强的能力，比如通信录、日历、短信、振动、传感器和诸如近场通信等新技术。我们在PhoneGap应用中只是对相机的使用稍稍触及一点皮毛。

如果浏览器开始支持设备API，至少从理论上讲会使很多方面成为可能。

什么是设备API？

原先，设备API这个词是指在移动设备上可用而在桌面上不可用的一组特定功能。手机相机就是这方面很常用的一个例子。

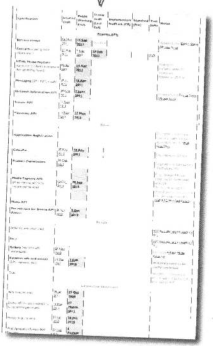
很多特定的功能原先似乎只有移动设备上才有，但是现在已经全面面向各种浏览器，从为日历增加事件到访问系统信息（如CPU使用情况和网速）都能提供。

尽管范围变得更宽，但重点还是一样，仍然是允许浏览器访问成就了移动设备的那些特性。

W3C®

WAC

可以从公共路线图（http://w3.org/2009/dap/）跟踪W3C设备API工作组的进展。



DAP和WAC设置的标准

引领设备API标准化的两大组织是W3C的设备API工作组（Device APIs Working Group，DAP）和大规模应用社区（Wholesale Applications Community，WAC）。

你可能比较熟悉W3C，因为它为HTML和CSS设置了标准，不过可能还没有听说过WAC。

WAC最初是由网络经营厂家和手持设备制造商发起的，着力为应用开发创建一个公共平台。他们选择了HTML5，并结合了两个现有的标准形成WAC 2.0标准。

WAC和DAP之间存在大量重叠，包括他们所写的规范，以及参加组织的成员。实际上，他们的协作多于竞争。

听起来很不错。那么为什么不在书中介绍设备API呢？

还没有太多浏览器支持设备API。很多标准还处于早期开发阶段。

希望到我们写这本书第二版的时候，这些标准已经很常见，那时也许我们需要用好几章专门介绍利用设备API可以做到哪些酷炫的事情。

#5 应用商店和分发

Apple的"App Store"有专有的商标，所以现在只有它的应用商店可以称为"App Store"，而我们不能随便使用。因此从现在开始我们都称作"应用商店"（application store）。

你已经构建了不错的Web应用，把它包装在PhoneGap中，希望它成为畅销应用。怎么能在应用商店里大赚一笔呢？

事实上：大多数应用都会赔钱

这并不表示你的应用不会成为例外，不过确实说明对于能不能让你致富还是要现实一点。

另外，应用的意义可能体现在很多方面，特别是那些并不依靠出售应用来谋生的公司，他们只是需要利用应用来扩大已有的业务。



每个人都有一个商店……

现在的商店实在太多了。无线行业联合会（Wireless Industry Partnership，WIP)维护了一个数据库，其中包括它能找到的所有应用商店。根据最新统计，现在有120个不同的应用商店。

可以在http://wipconnector.com/appstores找到WIP应用商店数据库。

每个商店有不同的条款和规则，推广应用的方式更是多种多样。选择实在是太多了。

……现在又加入了Web应用

最新趋势是所谓的HTML5应用商店。GoogleChrome、Mozilla和很多较小的公司都为基于Web的应用销售创建了新的应用商店。

一些非专业建议

我们并不是应用商店专家。你要到别处看看有没有更好的指导。不过可以告诉你三件事：

1. **要研究市场和重要的应用商店。**
   了解你的竞争对手都在做什么。对于你将加入的应用商店以及它更愿意推广哪些类型的应用，你要有更多了解。先做好功课。
2. **准备好一个传统的市场营销计划。**
   不要以为被Apple看重就大获成功了。这不是全部。只是构建应用还不够。
3. **要让你的应用感觉完全针对于用户的平台。**
   也许你使用了跨平台工具来完成应用的构建，但这并不表示Android用户了解一个iPhone应用如何工作。要让你的应用感觉就是针对用户平台构建的。

#6 RESS：响应式设计＋服务器端组件（REsponsive design + Server-Side）

长远来看，我们并不认为响应式Web设计或设备检测会变得重要得让其他技术完全消失。如果要说点什么，只能说我们认为这些技术的组合在很多情况下会很有意义。

Luke Wroblewski为这个组合（响应式设计加服务器端测试）给出了一个名字，他把它叫做RESS。

Luke所称的RESS技术与Yiibu.com的所有者Bryan和Stephanie Rieger采用的一些基础技术并没有太大差别。

Bryan和Stephanie使用设备检测对浏览器的能力做出最准确的猜测，然后使用客户端JavaScript来验证设备检测是正确的，如果不正确则修正可能的问题。JavaScript还会设置一个cookie，为服务器提供信息，可以对它之前做出的错误假设进行更正。

http://www.lukew.com/ff/entry.asp?1392



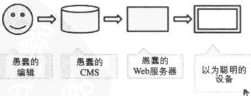
可以在http://browser.nokia.com实际了解Bryan和Stephanie的工作。



可以在http://www.slideshare.net/yiibu和http://yiibuo.com更多地了解他们的方法。

RESS的未来

很难说RESS这个名字能不能一直保留，不过将响应式Web设计和服务器端技术结合起来，这个概念对我们来说确实很有意义。如果没有这个组合，我们不难想起移动技术专家Jon Arnes Sæterås 最近给出的一个示意图。



http://mpulp.mobi/2011/05/next-steps-of-responsive-web-design/

我们的目标应该是确保每一部分都尽可能聪明。类似RESS的解决方案试图找到一种方法，能够使客户和服务器共同分担责任，提供绝妙的用户体验。

ii 建立Web服务器环境

就要开始了



没有“Web”，“移动Web”无从谈起。对于这一点绝对没有异议。如果要完成移动Web开发，肯定需要有一个Web服务器。这远不只是完成本书中的练习那么简单。你要把你的Web托管内容放在某个地方，不论是使用一个第三方商业Web托管服务，还是企业级数据中心，或者甚至是你自己的计算机。在这个附录中，我们将带领你在你的计算机上逐步建立一个本地Web服务器，并使用免费开源的软件运行PHP。

你要做的准备

要完成本书中的练习，你需要做几件事。不用担心。没有你想象中那么糟，我们会告诉你怎么做。

1. **要提供Web页面文档（如HTML页面）和资源（图像、CSS、JavaScript等）。**

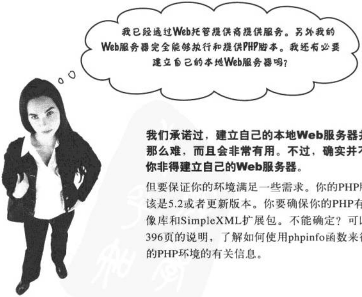
   这是Web服务器的根本，是不是？所以我们需要有一些Web服务器软件，并完成配置。

2. **需要执行服务器端PHP脚本。**

   这本书的练习使用了PHP（PHP: Hypertext Preprocessor）脚本语言来生成动态内容。尽管不需要你精通PHP，不过起码应该能够运行PHP。

3. **至少要对你的Web服务器有一定程度的管理访问权限。**

   对于Web服务器提供的目录，你要对这些目录有一定的控制权限，而且要能够移动文件。

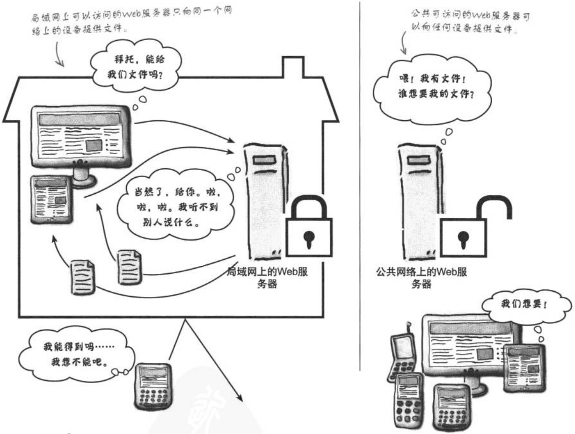


我们承诺过，建立自己的本地Web服务器并没有那么难，而且会非常有用。不过，确实并不要求你非得建立自己的Web服务器。

但要保证你的环境满足一些需求。你的PHP版本应该是5.2或者更新版本。你要确保你的PHP有GD图像库和SimpleXML扩展包。不能确定？可以看看396页的说明，了解如何使用phpinfo函数来得到你的PHP环境的有关信息。

只在本地可用

现在我们要带着你在你自己的计算机上建立一个本地Web服务器。



Watch it!

我们会跟随这个附录安装一些Web软件包，不过这些软件包并不面向公共可访问的生产环境。

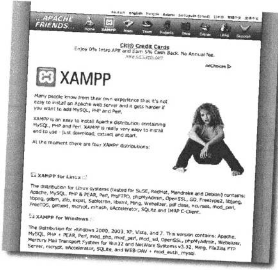
由于这些工具只是为了帮助你在自己的计算机上完成简化的本地Web开发，它们在安全方面还有一些瑕疵。不能在外部可访问的Web服务器上使用这些工具。一定要当心！

Windows和Linux：安装和配置XAMPP

下载Web服务器包软件

我们要使用XAMPP，这是一个易于使用的Web集成工具，其中包含Apache Web服务器软件和PHP，这两样都是我们需要的。它还提供了MySQL关系数据库管理系统（Relational Database Management System，RDBMS）和Perl，尽管这两个工具在这本书中没有直接用到，不过它们也很不错。

把浏览器指向www.apachefriends.org/en/xampp.html，并选择适当的平台链接（Windows或Linux）。在你的计算机上下载最新版本的XAMPP。



Windows环境下的安装

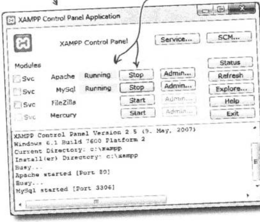
启动下载的文件，运行安装程序。会提示你为XAMPP选择一个安装目录。默认的c:\xampp\ 文件夹就很好。按照安装说明完成安装。

然后可以从Start（开始）→Programs（程序）→ XAMPP启动XAMPP。

没有吗：可以看看你的Start（开始）菜单的All Programs（所有程序）子菜单。

XAMPP控制面板。

可以在这里开始和停止你的Web服务器(Apache)。



启动XAMPP

Linux用户，你可以这样安装

下载软件包之后，打开一个终端窗口，执行以下命令。

你需要超级用户（root）访问权限才能安装XAMPP。

你的文件名可以稍有不同。

一旦安装XAMPP，使用这个命令来启动。

使用“stop”（而不是“start”）可以停止服务器。很显然，是不是？

```
File Edit Window Help MakeTuxProud
$ su
$ tar xvfz xamp-linux-1.7.7.tar.gz -C /opt

$ /opt/lampp/lampp start

Starting XAMPP 1.7.7......
LAMPP: Starting Apache......
LAMPP: Starting MySQL......
LAMPP started.

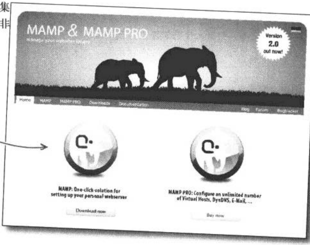
Ready. Apache and MySQL are running.
```

网上还有更多信息

要了解更多命令、配置选项、一般信息和排错说明，Windows用户可以在www.apachefriends.org/en/xampp-windows.html找到更多信息；如果你运行的是Linux，可以查看www.apachefriends.org/en/xampp-linux.html。

Mac用户：现在是MAMP时间

对于Mac，我们建议使用超级简单的Web软件集成工具MAMP，而且它很有意思。安装MAMP非常容易。

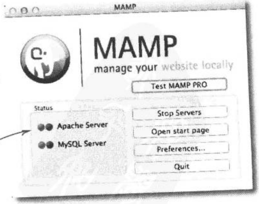


MAMP的常规版本就很不错。

有点像记录日志

访问http://mamp.info，下载软件包。双击装载的磁盘映像，把MAMP应用拖到Applications文件夹里。

启动MAMP应用时，可以开始和停止你的Apache和MySQL服务。对我们来说，我们更关心Apache。要完成本书中的练习，必须保证这个服务器一直在运行。



可以通过MAMP控制面板开始和停止你的Web服务。

确保Apache服务器正在运行。

确保有正确的端口

在Mac上……

Apache Web服务器已经预打包了Mac OS X的最新版本。这个服务器在端口80运行，这是默认的HTTP端口。在浏览器中访问http://localhost，就能访问你的Mac的默认Web服务器（如果它在运行）。正像我们前面所说，端口80是默认的HTTP端口，所以http://localhost和http://localhost:80都可以（也就是说，端口号不是必要的）。

可以在System Preferences（系统首选项）> Sharing（共享）面板中打开或关闭Web Sharing（Web共享），来控制Mac的Web服务器。

我们安装了MAMP，因为还需要PHP。由于大多数Mac上已经有一个Web服务器（MAMP）在端口8888上运行。这说明，可以通过http://localhost:8888访问MAMP Web服务器。如果需要，可以很容易地在MAMP首选项中改变端口号。

……在Windows和Linux上

默认地，XAMPP会在端口80运行。这说明，如果你在这个端口上运行了另一个Web服务器，就会有问题。在Windows上，可能需要先检查，确保没有运行Windows IIS Web服务器。



1. 通过Start（开始）→ Control Panel（控制面板）→ System and Security（系统和安全）→ Administrative Tools（管理工具）→ Internet Information Services (IIS) Manager，打开IIS Manager。

2. 在Actions（动作）面板中，点击Stop（停止）。这会停止IIS Web服务器。

如果你清楚自己在做什么，还可以配置Apache在一个不同的端口上运行。

访问你的Web服务器

你可能想在一个真正的移动设备上测试这本书中的一些例子，把你的计算机用作Web服务器。这是可以的，即使你的Web服务器只能在本地使用。按下面的要求确保你已经做好准备：

1 确定你的计算机的当前IP地址。

localhost都是相对的。它指示你目前所在的计算机。所以如果想从另一个设备（移动设备或其他设备）访问localhost，会指向设备自身。

要查找你的计算机的当前IP地址，在Mac或Linux上，可以从一个终端窗口运行命令ifconfig。Windows用户可以在开始菜单的搜索框中输入cmd，这会得到一个命令行工具，然后在命令行工具中输入ipconfig来运行ipconfig命令。

ifconfig（或ipconfig）会输出你的计算机各个网卡的有关信息。查找一个以eth或en（Linux或Mac）或IP4（Windows）开头的网卡，找到你的以太网卡的有关信息。可以在其中看到你当前的IP地址。

内部IP地址通常以10、127或192开头。

Watch it!

要找到你的计算机的IP地址，而不是外部IP地址。

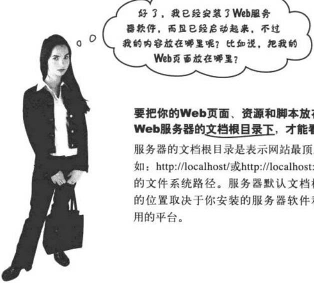
http://whatismyipaddress.com之类的工具很方便，可以帮助你很快找到你的外部IP地址。不过这与你的计算机的IP地址并不一样。如果你在一个使用modem和/或路由器连接到互联网的网络上，在这些网站上你看到的IP地址就是你的modem或路由器的外部IP地址。用这个IP地址（很可能）无法访问你的计算机的Web服务器。

2 确保你的设备在同一个网络上。

要记住，如果你有一个内部IP地址，这个IP地址只能在同一个网络内部访问。需要访问你的Web服务器的设备都必须连接到同一个网络（对于移动设备，通常要通过WiFi）。

不要忘记，如果你的端口号不是80，还要增加端口号，例如，http://10.0.0.2:8888。

作为Windows用户，是不是还有麻烦？查看这个Apache Friends支持论坛的贴子，其中对访问XAMPP服务器可能遇到的问题提供了解决办法：http://bit.ly/sG3Qa0。



要把你的Web页面、资源和脚本放在你的Web服务器的文档根目录下，才能看到。

服务器的文档根目录是表示网站最顶层（例如：http://localhost/或http://localhost:8888/）的文件系统路径。服务器默认文档根目录的位置取决于你安装的服务器软件和所使用的平台。

得到文档根目录

对于运行MAMP的Mac用户，默认的文档根目录是/Applications/MAMP/htdocs。可以保持这个路径不变，或者，如果你想改变这个位置，可以在Preferences（首选项）的Apache标签页中编辑。

对于使用XAMPP的Windows用户，文档根目录是\xampp\htdocs。对于使用XAMPP的Linux用户，文档根目录在/opt/xampp/htdocs。

需要把希望从Web服务器访问的所有文件和目录都放在你的文档根目录中。

phpInfo，拜托了！

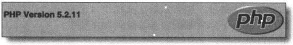
要验证正在运行的PHP的版本、选项、扩展包以及其他繁杂的细节，可以在你的文档根目录中创建一个名为test.php的文件，从而能通过http://localhost/test.php访问这个文件。

在这个文件中，只需要增加这样一行代码：

```
<?php phpinfo(); ?>
```

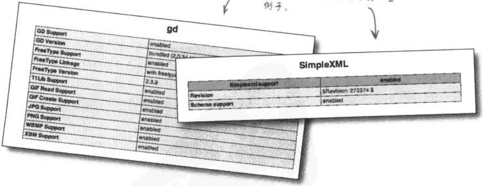
如果你的端口号不是80，还要增加端口号（例如，http://localhost:8888/test.php）。

保存这个文件，并在Web浏览器中访问http://localhost/test.php。这会告诉你PHP是否在运行，以及有关的所有信息。



页面最上面指示正在运行的PHP的版本。

在页面上继续向下检查gd和SimpleXML部分。我们需要这些特性来完成这本书中的一些例子。



搜寻设备



要揭开设备检测的神秘面纱，第一步要做些调查工作。优秀的侦探都知道，要解开谜团，必须搜集线索，询问目击证人。首先，我们要找出行动的策划者：WURFL PHP API。接下来，追踪行动者，将数以千计设备的能力信息放在一个XML数据文件中。不过还需要做些协调，才能让它们协作解决整件事情，所以我们要对配置稍做调整，认真做些笔记。

谁是策划者？

我们已经跟踪到WURFL API及其数据。下面是我们的计划。

- [ ] 下载和安装WURFL的“策划者”：PHP API。
- [ ] 下载和安装“行动者”：WURFL XML设备数据。
- [ ] 做一些微小的配置调整，并做一些记录。

WURFL (Wireless Universal Resource File，无线通用资源文件）是一个XML数据文件，包含移动设备及其浏览器的大量能力和特性支持数据。另外还有一个API，用于与这个数据文件交互。我们也需要这个API！

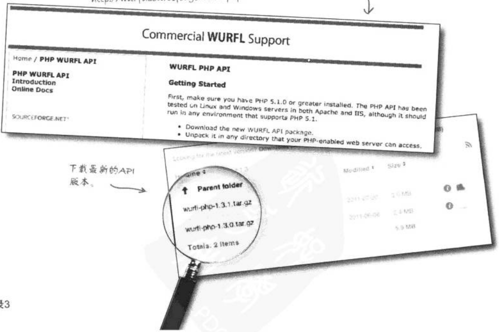
请和我们一起念“WURRRR-full”。

下载API

访问WURFL PHP API项目页面（http://wurfl.sourceforge.net/nphp/）。找到下载WURFL API包的链接，从下载页面找到最新版本。

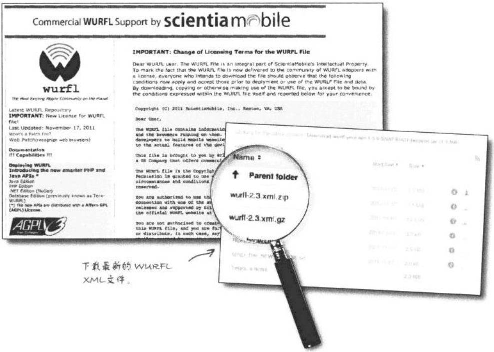
这就是PHP API。解压缩下载的文件，把得到目录入在你的计算机上的某个位置。具体放在哪里并不重要，只要你记住它在哪里就行了。

http://wurfl.sourceforge.net/nphp/



谁来行动?

http://wurfl.sourceforge.net/wurfl_download.php



下载最新的WURFL XML文件。

PHP API很聪明，不过如果没有它“身强力壮”的追随者（WURFL设备数据本身），它什么也做不了。

要得到最新的WURFL设备数据XML文件，可以访问http://wurfl.sourceforge.net/wurfl_download.php。你要先阅读并同意许可条款，才能下载这个文件。

一旦下载了文件，解压缩，把它重命名为wurfl.xml，并移动到WURFL PHP API安装目录的examples/resources/目录中（这个目录应该已经存在）。

下载文件的文件名中可能还包含一个版本号。把它重命名为wurfl.xml。

让二者协作

我们进展很顺利。现在需要对WURFL配置文件做些小小的修改，使我们的API和数据可以协同工作。

编辑WURFL API安装目录中的文件examples/resources/wurfl-config.xml。需要对其中一行代码做简单的修改。

- [x] 下载和安装WURFL的“策划者”：PHP API。
- [x] 下载和安装“行动者”：WURFL XML设备数据。
- [ ] 做一些微小的配置调整，并做一些记录。

把它……

```
<wurfl-config>
    <wurfl>
        <main-file>wurfl.zip</main-file>
        <patches>
            <patch>web_browsers_patch.xml</patch>
        </patches>
    </wurfl>
```

wurfl-config.xml

……改为：

```
<wurfl-config>
    <wurfl>
        <main-file>wurfl.xml</main-file>
        <patches>
            <patch>web_browsers_patch.xml</patch>
        </patches>
    </wurfl>
```

wurfl-config.xml

现在WURFL API就能知道在哪里查找数据。

做一点文件系统“家务活”

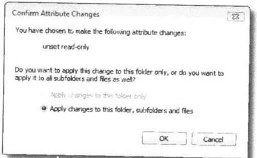
下面确保WURFL能得到它需要的所有“毛巾”。按照我们对WURFL的配置，它会使用文件系统缓存。所以我们要确保文件系统已经对它的到来做好准备。

首先，验证WURFL API目录中存在examples/resources/storage目录，它有两个子目录cache和persistence。

如果还没有这些目录，现在必须创建。

你的Web服务器必须能够向这些目录写文件，WURFL才能正常工作。对于MAMP（Mac），这可以自动做到，因为默认地，Apache Web服务器会作为你自己的用户运行。所以，如果你可以写这些文件，Apache也同样可以。

不过，在Windows和Linux上就不同了。在Windows上，右键点击资源管理器中的storage文件夹，如果这个文件夹的“只读”属性选中，则取消选中。你会看到一个确认对话框，与这里显示的类似。另外，没错，你可能要对子文件夹和文件做一些修改。



Linux用户可以使用chmod和chown，直到他们认为设置合适为止！

Relax

应该会很顺利。不过，如果出了问题，可以按PHP的说明来处理……

如果这些文件系统家务活还没有把所有灰尘打印干净，或者WURFL配置中还有些问题。通常PHP提醒、警告和错误会提示你发生了什么。

如果处理PHP或WURFL例子时得到一个空白屏幕，说明你可能没有打开PHP错误显示。在出问题的PHP文件所在的目录中创建一个名为.htaccess的文件（如果这个文件还不存在），并增加下面这行代码：

```
php_flag display_errors on
```

记笔记！

开始做第5章的练习之前，需要记下你的系统上的几个路径。

WURFL PHP API代码路径

Mac和Linux路径与这些例子很相似。Windows用户则要使用Windows兼容的路径（要用反斜线，还要注意另外一些方面）。

第一个是WURFL PHP API代码位置的完整路径。它位于WURFL PHP API目录的WURFL子目录中。

例如，如果你有1.3.1版本的API，并把它安装在一个（虚构的）/awesome目录下，你的路径就会是这样：

取决于你在哪里安装API，这个路径会有所不同。

对于第5章中的WURFL_DIR值，你要用你自己的路径值。

```
/awesome/wurfl-php-1.3.1/WURFL/
```

如果你有一个新版本的API，这一部分会不同。

不要忘记WURFL目录。具体的代码会放在这里（而不是放在顶层目录中）。

资源路径

这是WURFL PHP API目录中examples/resources/目录的完整路径。例如：

```
/awesome/wurfl-php-1.3.1/examples/resources/
```

对于第5章中的RESOURCES_DIR值，你要使用你自己的路径值。

你的WURFL数据都放在这里。

结案

- ☑ 下载和安装WURFL的“策划者”：PHP API。
- ☑ 下载和安装“行动者”：WURFL XML设备数据。
- ☑ 做一些微小的配置调整，并做一些记录。

iv 安装Android SDK和工具

完善环境

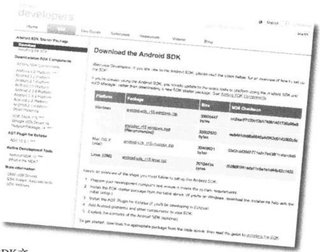


要成为测试原生Android应用的主人，必须当心环境。要把你的计算机变成一个条件宜人的小环境，在这里你可以在虚拟（模拟）设备或真实设备上“放牧”Android应用。为了让你成为合格的牧羊人，管理好这些Android小羊，我们会告诉你如何下载Android软件开发包（Android Software Development Kit，SDK），如何安装一些平台工具，如何创建一些虚拟设备，以及如何安装和卸载应用。

下载Android SDK

动手做！

访问http://developer.android.com/sdk，找到对应你的操作系统的下载链接。



http://developer.android.com/sdk

下载并解压缩SDK软件。把得到的Android SDK文件夹放在你的计算机上存放应用的位置。

Windows用户会得到一个很方便的安装工具，可以大大简化这个过程。

不错！现在你已经完成了SDK的安装（很容易，是不是）。

现在需要向这个SDK增加另外一些包和工具，来完成我们的工作。

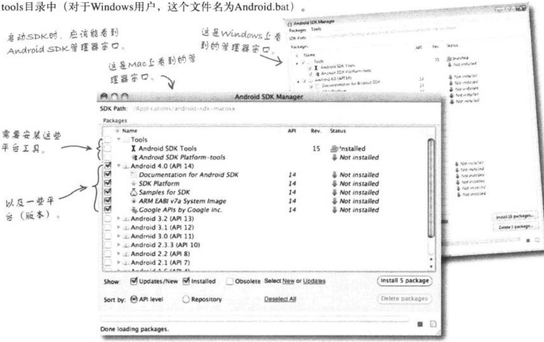
建立Android SDK环境的步骤

- [x] 对应你的计算机上的操作系统，下载并安装适当的Android SDK。 ← 已经完成。
- [ ] 安装一些必要的平台工具和一些Android平台（API版本）。
- [ ] 创建一个Android虚拟设备(AVD)。可以在这些模拟器上运行Android应用。
- [ ] 学习如何在模拟器和设备上安装和卸载Android应用。
- [ ] 配置PATH设置，从而可以更方便地运行Android工具。 ← 可选，不过完成这一步后会很方便。

找到最合适的工具

安装SDK很容易，不过这还只是开始，我们的工作还远没有结束。现在需要增加一些Android平台和工具。

找到并启动android执行文件。它应该在Android SDK文件夹的tools目录中（对于Windows用户，这个文件名为Android.bat）。



安装平台和工具

我们希望安装一些平台（即Android API版本）和工具。为此，选中想要安装的项旁边的复选框。然后继续：

SDK的各个版本不一定提供所有Android版本。

Android 3.0（即Honeycomb）是一个只面向平板电脑的Android版本。如果你愿意，也可以安装这个版本，不过这个版本的模拟器上可能会有不同的表现。

1. **安装两个或三个最新的Android平台。**

   写这本书时，4.0是Android的最新版本。我们还安装了2.2和2.3.3，用来提供另外一些可供测试的版本。如果你愿意，还可以安装更多的版本。

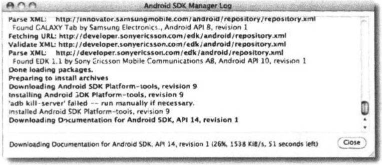
2. **安装"Tools"包。**

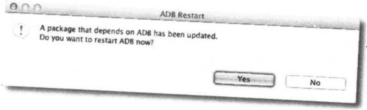
   我们特别需要一个名为adb的工具，它在Android SDK Platform Tools包中。选中Tools包文件夹旁边的复选框。

单击Install按钮，开始煮咖啡

一旦选择了平台版本和Tools包，单击Install按钮，开始安装。这会花一点时间，具体地，某些系统可能需要最多10到20分钟的时间，所以你可以趁这个功夫喝点咖啡休息一会儿。

可能会看到一个与这里类似的安装日志。





安装过程中，可能会看到类似这样的一个提示框。点击Yes然后继续。

- [x] 对应你的计算机上的操作系统，下载并安装适当的Android SDK。
- [x] 安装一些必要的平台工具和一些Android平台（API版本）。
- [ ] 创建一个Android虚拟设备(AVDs)。可以在这些模拟器上运行Android应用。
- [ ] 学习如何在模拟器和设备上安装和卸载Android应用。
- [ ] 配置PATH设置，从而可以更方便地运行Android工具。

需要建立一些虚拟设备，用于测试。

虚拟设备梦？

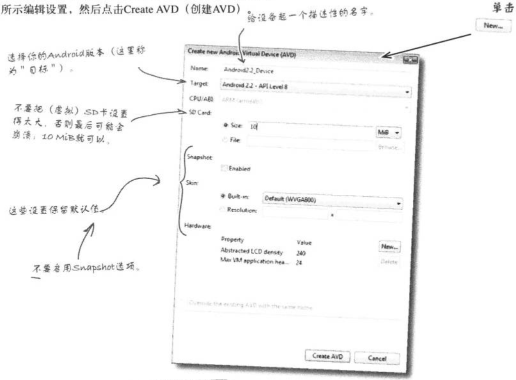
基础已经有了：现在我们需要一些模拟设备。在Android SDK中，选择Tools → Manage AVDs菜单项。



我们需要增加一些设备。单击这个New按钮，开始增加设备。

创建一个新的虚拟设备

在AVD管理器窗口中点击New按钮，会显示类似这样的一个窗口（这个截屏图截自Windows系统）。按照图中所示编辑设置，然后点击Create AVD（创建AVD）。





动手做！

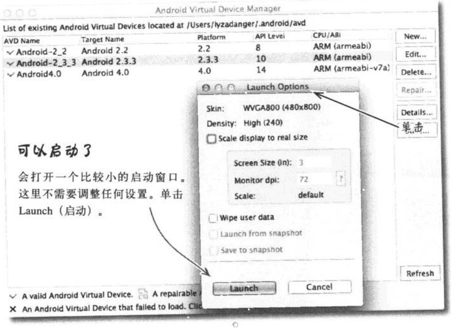
不要忘记，3.0版本只面向平板电脑，可能不适合你。

在不同的Android平台（API版本）上创建3个模拟设备。建议不要使用太老的版本，比如2.2之类的版本。

全速前进！

一旦创建了AVD，可以从AVD Manager启动这些虚拟设备（应该记得，可以通过Tools → Manage AVDs菜单项打开AVD Manager）。

要启动一个特定的模拟设备，可以在列表中选中这个设备，然后单击Start按钮。



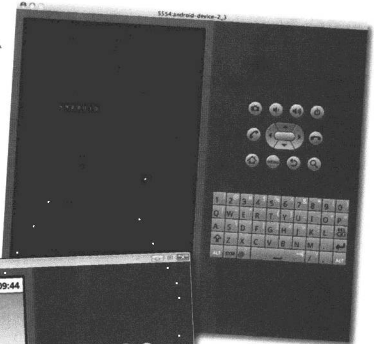
Relax

启动模拟Android设备可能要花些时间。

启动模拟器可能是一个很慢的过程，特别是第一次启动时需要的时间会比较长。耐心等一会儿，如果看起来时间很久也不要慌。

如果确实发现模拟器无法启动或者崩溃了，要反复检查确保SD卡没有设置得过大，另外要确保没有打开Snapshot选项。最后一点，对于某些版本，有时模拟器在一些系统上根本不能工作。如果你发现确实启动有困难，可以考虑换一个不同的API版本。

Mac上启动的一个Android 2.3 AVD。





Windows上运行的一个Android 4.0 AVD。

应用安装，应用卸载

- [x] 对应你的计算机上的操作系统，下载并安装适当的Android SDK。
- [x] 安装一些必要的平台工具和一些Android平台（API版本）。
- [x] 创建一个Android虚拟设备(AVD)。可以在这些模拟器上运行Android应用。
- [ ] 学习如何在模拟器和设备上安装和卸载Android应用。
- [ ] 配置PATH设置，从而可以更方便地运行Android工具。

还记得吧？我们下载了Android SDK之后安装了Platform Tools包。这个包中包含一个程序adb，这是安装和卸载应用的关键，同时对于管理虚拟和真实的设备也非常重要。

安装和卸载应用的过程很简单：

1. 确保目标模拟器（AVD）完全加载并运行，或者你的Android设备已经连接到计算机的USB口。

2. 分别使用adb install或adb uninstall命令来安装或卸载指定的应用。

adb表示Android debug bridge（Android调试桥）。利用这个工具程序，我们可以在模拟器和设备上安装、卸载和调试Android应用。

安装

```
$ adb install MyApp.APK
```

App包文件

卸载

```
$ adb uninstall com.foo.myApp
```

App包

Android应用使用Java风格的包名，就像是逆序的域名。

我不知道从哪儿开始。我从哪里得到能安装的APK文件？在哪里输入adb install命令？



我们走得有点太急了！抱歉！

我们会告诉你在哪里找到一个可以安装的示例Android应用。不过，首先来了解如何运行这个adb命令。

路径查找

几乎可以确定你的PATH中不会包括Android SDK工具（包括adb）。所以，如果打开一个终端窗口，或者使用Window的命令行工具，想要直接运行abd，可能会得到一个错误消息。你的系统需要知道在哪里能找到adb才能正确地执行这个命令。

建议你更新用户的PATH，让它包含Android SDK工具的路径，这样你就能很容易地从命令行直接运行adb命令了。

$PATH环境变量是一个冒号分隔的文件系统路径列表，你在命令提示窗口中输入一个程序名时，操作系统会在$PATH指定的路径下查找可执行程序。

在Windows上，路径之间用分号分隔。

通常，默认地这包括很多目录，但不包括SDK所在的目录。

Relax

如果无法指定PATH，还有另外的办法。

可以在运行adb时提供完整路径（例如，/Applications/android-sdk/platform-tools/adb），或者可以把目录切换为platform-tools目录，然后运行./adb（需要有./）。

找到正确的PATH

Mac用户

在终端窗口中，输入以下命令。然后需要关闭这个终端窗口，再打开一个新的窗口，修改才能生效。

第一个命令会进入用户的主目录。

按照这里所示键入命令。注意冒号(:)和大括号({})。

这里一定要包含单引号（而且要放在导出符号前面）。

```
File Edit Window Help 9-9-0-6-9-4-7-X-B-7-1
$ cd ~
$ echo 'export PATH=${PATH}:/Applications/<·····>/platform-tools' >> .profile
```

第二个命令把SDK平台工具的路径追加到用户.profile文件中现有的PATH变量后面。

每次打开一个新的终端窗口时会执行.profile。

你需要更新这一部分，与Android SDK中platform-tools目录的路径一致。

Windows用户

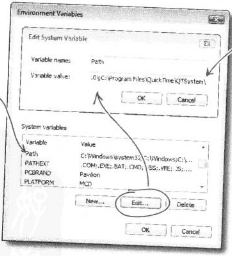
1. 在开始菜单中，右键单击“Computer”（计算机），然后进入“Properties”（属性）。点击“Advanced System Settings”（高级系统设置）。在显示的对话框中，选择“Advanced”（高级）标签页。

2. 在这个“Advanced”（高级）标签页中，单击“Environment Variables”（环境变量）按钮，然后单击“System Variables”（系统变量）部分中的Path变量。单击“Edit”（编辑）。

3. 在“Variable value”（变量值）域中，把光标放在当前值的末尾，输入一个分号(;)，然后输入Android SDK platform-tools文件夹的路径。路径最后不必再加一个分号。单击OK多次，逐个确认并关闭相应的对话框。

4. 需要注销，然后再次登录，这个修改才能生效。

把platform-tools文件夹的路径增加到当前path值的最后。



Linux用户，应该不需要向你介绍这一步怎么做吧。

试一试

好了，我们拭目以待的时刻终于到了。下面就来试一试，在一个模拟器上安装和卸载一个测试应用。

1. 从http://hf-mw.com/ch8/PhoneGap.apk下载PhoneGap Build演示Android应用。

2. 如果Android SDK还没有运行，将其启动。

3. 启动一个AVD（模拟的虚拟设备），为此进入Tools → Manage AVDs，从列表中选择一个AVD，并启动这个AVD。

4. 等待模拟设备完全加载。

5. 打开一个终端窗口或命令行工具。把目录切换到下载APK文件的位置。

6. 运行下面的命令：

这里假设你已经适当地设置了#PATH，即已经包含platform-tools目录。否则，可能需要为adb提供完整路径。

```
$ adb install PhoneGap.apk
```

7. 现在应该能够在你的模拟设备的应用屏幕上看到这个PhoneGap示例应用了。可以启动应用，试着运行看看。

8. 从命令行运行下面的命令，卸载这个应用。

```
$ adb uninstall com.phonegap.getting.started
```

这是PhoneGap "getting started" 应用的包ID。

there are no Dumb Questions

问： 什么是Android包文件？

答： Android包文件（Android Package File，APK）是一个打包的原生Android应用。它包含编译的应用代码，以及应用的各种资源。

APK是一种Java归档文件（Java Archive File，JAR）。如果你想知道一个特定的APK里有什么，通常可以把扩展名改为.zip，然后解压缩这个文件，这样就能"哄骗"计算机为你显示它的内容。看看吧！

问： 这个com.foo.myApp是什么？

答： Android应用包由Java风格的完全限定包名来标识。这些包名有点像逆序的域标记（也就是说，有点像倒过来的Web域名）。

adb工具还有很多很棒的特性，可以用来调试和检查应用和设备。

要想更多地了解adb还能做什么，可以从命令行输入adb help。

问： 我怎么知识我的应用的包名呢？

答： 第8章中，我们告诉过你如何为PhoneGap Build服务编辑一个XML配置文件，其中就设置了应用的包名。

问： adb怎么知道要在哪个设备上安装这个应用呢？

答： adb会在当前正在运行的模拟设备上安装或卸载应用。或者，如果你把一个真正的Android设备连接到计算机的USB口，它就会在那个真正的设备上安装或卸载应用。

问： 我能从模拟器或设备本身卸载一个应用吗？

答： 可以。可以像真实世界中一样卸载应用。进入设备的菜单，找到Manage Apps选项。然后可以找到并删除当前应用。

问： 我安装或卸载应用时得到了一个错误。

答： 在试图安装APK之前确保模拟器完全加载。可以运行命令adb devices来查看abd知道的设备或模拟器列表。如果你的模拟器或设备在正常运行，应该能看到一个相应的项。

安装或卸载时不能同时运行多个活动设备（包括真实设备和模拟设备），否则abd会不知所措。

最后，如果一个应用已经安装，再次安装之前必须先卸载这个应用。

问： 我有一个真正的Android设备。怎么安装和卸载应用呢？

答： 可以使用同样的命令，不过不用启动模拟器，只需要把你的设备插到计算机的USB口上。运行adb devices命令可以确认abd能看到你的设备。

哈！大功告成！

- [x] 对应你的计算机上的操作系统，下载并安装适当的Android SDK。
- [x] 安装一些必要的平台工具和一些Android平台（API版本）。
- [x] 创建一个Android虚拟设备(AVD)。可以在这些模拟器上运行Android应用。
- [x] 学习如何在模拟器和设备上安装和卸载Android应用。
- [x] 配置PATH设置，从而可以更方便地运行Android工具。

索引

数字

4G phones，4G手机, performance，性能 65

A

About Us page, Tartans Unlimited，About Us页面, Tartans Unlimited 271～272

accessing servers，访问服务器 394～395

accesskey attribute，accesskey属性 117

access keys，访问键
- attributes，属性 117, 118
  - scrolling，滚动 119
- list numbering，列表编号 132～133
- markup validation，标记验证 123
- testing，测试 125
- troubleshooting，排错 122～125

AcedIt! Test Prep 152
- device class groupings，设备类分组 184～185
- red panic button，紧急按钮 153
  - testing，测试 174～175
- standalone website plans，独立网站计划 182

adb 411～413, 415

addColor() function，addColor()函数 277

Add This Color button, adding，Add This Color按钮，增加 276

Adobe 318

Adobe Flash 35

Affero General Public License v3 (AGPL) 158

AGPL (Affero General Public License v3) 158

AJAX
- browser support，浏览器支持 281
- loading Web pages，加载Web页面 236

A List Apart article，A List Apart文章 374

analytics, device support，分析，设备支持 145

Android 3
- apps 403
  - installing，安装 415
  - installing with adb，用adb安装 411～413
  - testing，测试 414
  - uninstalling，卸载 415
  - uninstalling with adb，用adb卸载 411～413
- PhoneGap 319
  - creating native apps，创建原生应用 322～327
  - downloading APK (Android package)，下载APK（Android包） 328
- SDK (software development kit)，SDK（软件开发包） 319, 329, 403
  - adb 411～413, 415
  - adding emulated devices，增加模拟设备 407～408
  - downloading，下载 404
  - installing，安装 320, 404
  - launching emulated devices，启动模拟设备 409～410
  - platforms/tools，平台/工具 405～406
- testing strings，测试串 189

Android package (APK)，Android包（APK） 415
- downloading，下载 328

animation, jQuery Mobile，动画, jQuery Mobile 245

Apache, content-type 285

Apachefriends.org 390～391

Apache Friends Support Forum 394

APIs
- device，设备 384
- JavaScript, PhoneGap 345～349
- localStorage 333, 334～338
  - browser support，浏览器支持 343
  - client-side feature detection，客户端特性检测 339
  - data storage，数据存储 343
  - JavaScript magnets，JavaScript磁贴 336～338
  - keys，键 343

toggle methods，toggle方法 341～342
updating refreshTartans function，更新refreshTartans函数 352
mediaCapture 348～353
cameras，相机 344
PHP 156
capability values，能力值 173
matches，匹配 212～215
RIS (reduction-in-string) matching，RIS（串内归约）匹配 159
ScientiaMobile 158
Twitter 95
W3C mediaCapture 344
World Wide Web Consortium (W3C) Geolocation 298
WURFL PHP 397
code，代码 402
configuring，配置 400
filesystem access，文件系统访问 401
resource path，资源路径 402
WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件） 159, 213

APK (Android package)，APK（Android包） 415
downloading，下载 328

appCache 284.参见cache manifest
browser support，浏览器支持 296
localStorage 296
testing，测试 295
troubleshooting，排错 289, 291

app.js file，app.js文件 336～352
testing，测试 353～355

Apple, roles in changing mobile phone usage，Apple,改变移动手机用途中的地位 2～3

application cache.参见 appCache
definition，定义 284
troubleshooting，排错 289

application programming interfaces. 参见 APIs

application stores，应用商店 385

archaeology, user agents，考古，用户代理 98～99

asynchronous requests，异步请求 221

attributes，属性
accesskey 117
data-* 231～232, 236
data-icon 250
data-inset 233
data-role 232
creating navbars，创建导航条 249～250

manifest，清单 285, 286
moving Google Maps iframe attributes to CSS，将Google Maps iframe属性映射到CSS 80
removing from JavaScript，从JavaScript删除 81
src
CSS 68
<img> tag，<img>标记 68～69

audience considerations，用户考虑
case studies，案例研究 146～149
determining devices to support，确定要支持的设备 142～145

B

backend enhancements, super mobile web app enhancements，后端增强，超级移动Web应用增强 280～283

bearer capability group，承载能力组 176

BlackBerry OS, debugging，BlackBerry OS,调试 381

BlackBerry Playbook, debugging，BlackBerry Playbook,调试 381

Blaze 46
Mobitest 47～53
test result variations，测试结果变化 65

bookmarklets, installing，bookmarklets,安装 85

Bowling Boxers 147～150, 149～150

breakpoints (content)，断点（内容） 89

breaks (content)，中断（内容） 85～90

browsers，浏览器
AJAX 281
application cache，应用缓存 284
camera access，相机访问 316
Chrome
testing cache manifests，测试缓存清单 286
troubleshooting cache manifest，缓存清单排错 291
device support，设备支持 140, 144
downloading CSS images，下载CSS图像 71
installing bookmarklets，安装bookmarklets 85
Internet Explorer
conditional comments，条件注释 62～63
mobile media queries，移动媒体查询 62～65
queries media，查询媒体 65
testing media queries in，测试媒体查询 61～64
linking phone numbers，链接电话号码 178
markup validation，标记验证 117
mobile phones，移动手机 3
mobile web browsers，移动Web浏览器 6

Opera Mini 110
simulator，仿真器 111
Opera Mobile 110
plug-ins，插件 45
redirecting，重定向 104～109
Safari 110
testing cache manifests，测试缓存清单 286
troubleshooting cache manifest，缓存清单排错 291
sniffing，嗅探 116
support，支持
cache manifest，缓存清单 296
caniuse.com 382
client-side feature detection，客户端特性检测 339
geolocation，地理定位 303～308
localStorage 343
testing，测试 383
user agents，用户代理 98～103, 102～103
W3C Geolocation API，W3C地理定位API 299～300
WebKit 189
XHR 281
XHTML-Mobile Profile 116

browserscope.org 383

buildAddButton() function，buildAddButton()函数 277

build.php 281

buttons，按钮
adding to headers，增加到页眉 262～263
adding to navbars，增加到导航条 249～250
Add This Color, adding，Add This Color, 增加 276
Make It! submit button，Make It!提交按钮 276

C

cache manifest，缓存清单
browser support，浏览器支持 296
CACHE section，CACHE部分 289, 290
wildcards，通配符 296
content-type 285
creating，创建 285～296
changing resource list，修改资源列表 290～296
.htaccess files，.htaccess文件 287～288
syntax，语法 285
WebKit Web Inspector 286
definition，定义 284
FALLBACK section，FALLBACK部分 296
NETWORK section，NETWORK部分 289
wildcards，通配符 296
resources，资源
changing，修改 290～296
listing，列出 289
testing，测试 286, 295
troubleshooting，排错 289, 291, 292～293
URLs 285

CACHE section，CACHE部分 (cache manifest，缓存清单) 289, 290
wildcards，通配符 296

callback function, mediaCapture API，回调函数, mediaCapture API 349

cameras，相机
mediaCapture API 344
mobile browser access，移动浏览器访问 316

caniuse.com 382

capabilities，能力
groups，组
bearer，承载 176
device classes，设备类 181
retrieving capability values，获取能力值 171
testing devices in device class definitions，设备类定义中的测试设备 193～198
WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件） 170
creating device classes，创建设备类 187～189, 190～199

case studies, analyzing audience requirements，案例研究，分析用户需求 146～149

centers for testing mobile devices，测试移动设备的中心 375

Charles Proxy 65

charts，图表
pie，饼图 50
waterfall，瀑布图 48
analyzing image sizes，分析图像大小 54
HAR (HTTP Archive) 49～50
reading，读 51～52

Chrome
testing cache manifests，测试缓存清单 286
troubleshooting cache manifest，缓存清单排错 291

classes，类
device，设备 151, 186
creating with WURFL capabilities，用WURFL能力创建 190～199
definition of，定义 180
desktop，桌面 186
grouping capabilities，分组能力 181
grouping experiences，分组体验 183, 184～185
names，名字 189
numbers per project，每个项目编号 189
PHP API 212～215
study aid site (AcedIt! Test Prep site)，学习辅助网站

(AcedIt! Test Prep site) 201～209
testing，测试 200, 209
troubleshooting，排错 198
click event，click事件 277～278
client-side device detection，客户端设备检测 150
client-side feature detection，客户端特性检测 339
CMS (content management system)，CMS（内容管理系统）
Creature Comforts International，动物关怀国际组织 93～95
mobile devices and，移动设备 95
code，代码
jQuery Mobile 228～229
validation，验证 374
WURFL PHP API 402
color fields，颜色域 259～260
colors, adding with lists，颜色，用列表增加 258
color-size field combos，颜色-大小组合域
generating arbitrary combos，生成任意个组合域 276
removing，删除 275
color-size field pairs，颜色-大小域对 260
columns，栏
converting to fluid layout，转换为流式布局 29
fluid layouts，流式布局 37
com.foo.myApp 415
commands, ifconfig，命令，ifconfig 394
conditional comments，条件注释
Internet Explorer 62～63
media queries，媒体查询 63～65
config.php 280
configuration files, creating，配置文件，创建 162～168
connection speed, device support，连接速度，设备支持 144
content，内容
floating，浮动 60
future-friendly manifesto，未来友好宣言 366
overlap，重叠 83～89
reordering，重排序 59～60, 65
semantic markup，语义标记 57
content breakpoints，内容断点 89
content breaks, media queries，内容中断，媒体查询 83～88
content management system (CMS)，内容管理系统（CMS）
Creature Comforts International，动物关怀国际组织 93～95
mobile devices and，移动设备 95
content-type, cache manifest and，content-type,缓存清单 285
context switching, fluid layouts，上下文切换，流式布局 31～32
Creature Comforts International，动物关怀国际组织 92
CMS (content management system)，CMS（内容管理系统） 93～94, 95
infrastructure，基础设施 93
mobile device detection script，移动设备检测脚本 104～107
mobile website needs，移动网站需求 93～94
CSS
creating for mobile devices，为移动设备创建 14～21
analyzing CSS，分析CSS 16～17
analyzing site structure，分析网站结构 15
converting fixed layouts to fluid layouts，固定布局转换为流式布局 24～30, 34
identifying changes，找出改变 18
mobile magnets，移动磁贴 19～20
questions/answers，问题/答案 37
steps，步骤 19～23
updating styles.css file，更新styles.css文件 22
viewport <meta> tag，viewport <meta>标记 22
desktop structural，桌面结构 21
hiding JavaScript，隐藏JavaScript 53
images, downloading，图像，下载 71
media features，媒体特性 12
media queries，媒体查询 10～11, 13～14, 37
media types，媒体类型12～14
mobile devices and，移动设备 9
mobile media queries，移动媒体查询 61
moving iframe attributes to，iframe属性移动到 80
RWD (Responsive Web Design)，RWD（响应式Web设计） 38～40
shared structural，共享结构 21
Splendid Walrus website，非凡海象网站 15～19
src attribute，src属性 68
testing，测试 23
wireless，无线 135
CSS3 217
CSS Mobile Profile 2.0 (CSS-MP) 127～131
troubleshootin list numbering，列表编号排错 132～133
CSS Mobile Profile (CSS-MP) 134
CSS-MP (CSS Mobile Profile) 134
CSS-MP (CSS Mobile Profile 2.0) 127～131
troubleshooting list numbering，列表编号排错 132～133
css/styles.css, CACHE section，css/styles.css, CACHE 部分（cache manifest，缓存清单） 290

CustomDevice object，CustomDevice对象 189
custom widgets, jQuery Mobile，定制部件, jQuery Mobile 275

D

DAP (Device APIs Working Group) 384
data，数据
  future-friendly manifesto，未来友好宣言 366
  WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件） 159
data-* attribute，data-*属性 231～232, 236
databases (device)，数据库（设备） 151, 154
  DetectRight 157～158
  DeviceAtlas 157～158
  device support，设备支持 145
  MobileAware 157～158
  Wireless Universal Resource FiLe 参见 WURFL
data-icon attribute，data-icon属性 250
data-inset attribute，data-inset属性 233
data-role attribute，data-role属性 232
  creating navbars，创建导航条 249～250
data storage，数据存储 334～338
  localStorage API 343
data types, future-friendly manifesto，数据类型，未来友好宣言 366
Debug Bridge. 参见 adb
debugging，调试
  BlackBerry OS 381
  BlackBerry Playbook 381
  iWebInspector 381
  Opera Dragonfly 381
  remote debugging，远程调试 376～381
  WebKit 381
  weinre 376～381
debug.phonegap.com 377, 381
demographics, device support，人口统计，设备支持 145
design，设计 71, 357
  determining which design to use，确定使用哪个设计 358～361
  fluid layouts compared to fixed layouts，流式布局与固定布局比较 24～30
  future-friendly manifesto，未来友好宣言 362～363
    flexibility，灵活性 364
    guidelines for usage，使用原则 366～370
    Tartan Hunt 365
  mobile-first responsive，移动优先响应 56, 57
    converting into mobile-first design，转换为移动优先设计 58～59
    converting Responsive Web Design into，响应式Web设计转换为 58
    media queries，媒体查询 61
    widgets，部件 79～81
  progressive enhancement，渐进增强 57, 141
  RWD (Responsive Web Design)，RWD（响应式Web设计） 1
    context switching，上下文切换 31～32
    CSS media queries，CSS媒体查询 10～11, 13～14
    fluid layouts，流式布局 24～30
    images，图像 10～11
    layouts，布局 10～11
    media，媒体 10～11
desktop device class，桌面设备类 186
desktop structural CSS，桌面结构CSS 21
DetectRight 157～158
Device Anywhere 375
device APIs 384
Device APIs Working Group (DAP) 384
DeviceAtlas 157～158
device classes，设备类 151, 186
  creating with WURFL capabilities，用WURFL能力创建 187～199
  definition of，定义 180
  desktop，桌面 186
  grouping capabilities，分组能力 181
  grouping experiences，分组体验 183～185
  names，名字 189
  numbers per project，每个项目编号 189
  PHP API 212～215
  study aid site mockup (AcedIt! Test Prep site)，学习辅助模拟网站(AcedIt! Test Prep网站) 201～209
  testing，测试 193～198, 200, 209, 213
  troubleshooting，排错 198
device_classes.php file，device_classes.php文件 200
device databases，设备数据库 151, 154
  device support，设备支持 145
  Wireless Universal Resource FiLe. 参见 WURFL
device.php file，device.php文件 163～169
  adding tests to，增加测试 177
deviceready event，deviceready事件 345～346
devices，设备 6

A List Apart article，A List Apart文章 374
CMS (content management system)，CMS（内容管理系统） 95
creating CSS for，创建CSS 14～21
analyzing CSS，分析CSS 16～17
analyzing site structure，分析网站结构 15
converting fixed layouts to fluid layouts，固定布局转换为流式布局 24～30, 34
identifying changes，找出改变 18
mobile magnets，移动磁贴 19～20
questions/answers，问题/答案 37
steps，步骤 19～23
updating styles.css file，更新styles.css文件 22
viewport <meta> tag，viewport <meta>标记 22
detecting with scripts，用脚本检测 104～107
detection，检测 97
emulated，模拟
installing in Android SDK，Android SDK中安装 407～408
launching，启动 409～410
frames，帧 117
future-friendly manifesto，未来友好宣言 366
higher_mobile 186
installing Android SDK，安装Android SDK 329
native applications，原生应用 316
remote testing，远程测试 375
RWD (Responsive Web Design), converting fixed layouts to fluid layouts，RWD（响应式Web设计），固定布局转换为流式布局 24～30
scrolling on，滚动 119～125
sharing，共享 374
simpler_mobile 186
support，支持 137, 139
analytics，分析 145
audience considerations，用户考虑 142～145
browser features，浏览器特性 144
browsers，浏览器 140
connection speed，连接速度 144
customer considerations，客户考虑 150
demographics，人口统计 145
determining criteria/priorities，确定准则/优先级 138
geographic trends，地理趋势 145
hardware performance，硬件性能 144
jQuery Mobile 245
progressive enhancement，渐进增强 141
questions to ask，要问的问题 142～143
services，服务 144
stakeholders，投资人 144
unsupported devices，不支持的设备 140～141
targets，目标 117
testing，测试 139, 374～375
WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件）
instantiating，实例化 172
devices databases，设备数据库
DetectRight 157～158
DeviceAtlas 157～158
MobileAware 157～158
WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件）
capabilities，能力 156～157
dialogs, jQuery Mobile，对话框, jQuery Mobile 306
disabling scaling，禁用缩放 73
div#main content，div#main内容 37
div#points content，div#points内容 37
<div> tag，<div>标记 75
adding links，增加链接 75～76
DOCTYPE, definition of，DOCTYPE,定义 116
DOCTYPEs (HTML5), converting to XHTML-Mobile Profile，DOCTYPEs (HTML5), 转换为XHTML-Mobile Profile 114～115
document type declarations (DTDs)，文档类型声明（DDT） 116
domains, determining where files come from，域，确定文件来自哪里 52
downloading，下载
Android package (APK)，Android包（APK） 328
Android SDK (software development kit)，Android SDK（软件开发包） 404
CSS images，CSS图像 71
images, media queries，图像，媒体查询 66
resources, waterfall charts，资源，瀑布图 48～50
WURFL PHP API 398
XAMPP 390
XML data file，XML数据文件 399
DTDs (document type declarations)，DTD（文档类型声明） 116

E

ECMAScript Mobile Profile 135～136
EDGE network, 4G phones，EDGE网络，4G手机 65
em-based fonts, sizes，基于em的字体，大小 37
emulated devices (Android)，模拟设备(Android)
adding to Android SDK，增加到Android SDK 407～408

launching，启动 409～410

emulators，模拟器 374

installing Android SDK，安装 Android SDK 329

en-us 103

event_list.inc file，event_list.inc文件 309

events，事件

change 277～278

click 277～278

deviceready 345～346

JavaScript 365

pagecreate 275, 277～278

pageinit 275, 277～278

.ready 275

Tartanator 271～272

Events page, Tartanator，Events页面，花格布爱好者应用 271～272

explore page，资源管理页面

creating ScientiaMobile，创建ScientiaMobile 160～166

testing，测试 169

expressions (regular), mobile device detection script，表达式（正则），移动设备检测脚本 107

extensions, Google Chrome，扩展，Google Chrome 85

F

facebookexternalhit user agent，facebookexternalhit用户代理 213

FALLBACK section，FALLBACK部分 (cache manifest，缓存清单) 296

feature phones compared to smartphones，功能手机与智能手机对比 113, 120

fields，域

adding to forms，增加到表单 257

color-size field pairs，颜色-大小域对 260

size，大小 260

xhtml_make_phone_call_string 176

files，文件

app.js 336～352

testing，测试 353～355

build.php 281

config.php 280

configuration, creating，配置，创建 162～168

determining type，确定类型 52

device_classes.php 200

device.php, adding tests to，device.php, 增加测试 177

event_list.inc 309

generate.php 280, 281

.htaccess, creating cache manifest，.htaccess,创建缓存清单 287～288

image.php 280

index.html, Splendid Walrus website，index.html,非凡海象网站 15

index_mobile_.html, converting to XHTML-Mobile Profile，index_mobile_.html,转换为XHTML-Mobile Profile 114～118

index.php 209

index_solution.php 209

list.php 282

redirect.php 106

sizes，大小 52

HAR Viewer page pie charts，HAR Viewer页面饼图 50～51

styles.css

Splendid Walrus，非凡海象网站 16

testing，测试 133～134

updating，更新 22

tartans.html 240

tartans.php 282

XML data，XML数据 397

downloading，下载 399

filesystem access，WURFL PHP API，文件系统访问, WURFL PHP API 401

filters, list，过滤器，列表 244～249

Find Events page，查找事件页面

geolocation，地理定位 301, 309

JavaScript 301

Firebug 65

Firefox, troubleshooting cache manifest，Firefox,缓存清单排错 291

Firtman, Maximiliano 381

fixed layouts，固定布局

compared to fluid layouts，与流式布局对比 24～30

disadvantages of，缺点 26

Flexbox 65

flexibility, future-friendly manifesto，灵活性，未来友好宣言 364

Flexible Box Module 65

floating content，浮动内容 60

fluid images，流式图像 33, 54

fluid layouts，流式布局

advantages of，优点 27～28
columns，栏 37
compared to fixed layouts，与固定布局对比 24～30
context switching，上下文切换 31～32
images，图像 33

fluid media，流式媒体 35, 36

fonts
jQuery Mobile 230～231
sizes 35
em-based 37

footers
jQuery Mobile 237～239
swatch c 239
toolbars, adding to headers/footers with jQuery Mobile，工具条，用jQuery Mobile增加到页眉/页脚 249～252

forms，表单
geolocation，地理定位 301～308
HTML5 256～257
adding buttons to headers，向页眉增加按钮 262～263
adding fields，增加域 257
color fields，颜色域 259～260
color-size field pairs，颜色-大小域对 260
lists within lists，列表中的列表 258
testing，测试 263
super mobile web app enhancements，超级移动Web应用增强 274～277
Tartanator，花格布爱好者应用 253～261

frames，mobile devices and，帧, 移动设备 117

frameworks (mobile web apps)，框架（移动Web应用） 217, 225
disadvantages，缺点 225
jQuery Mobile
adding footer toolbars，增加页脚工具条 249～250
animation，动画 245
code components，代码组件 228～229
creating Tartanator page，创建花格布爱好者应用页面 228～239
creating tartans，创建花格布 240～245
custom widgets，定制部件 275
data-* attribute，data-* 属性 231～232, 236
device support，设备支持 245
dialogs，对话框 306
events，事件 275
footers，页脚 237～239
linking，链接 234
list dividers，列表分割条 244
list filters，列表过滤器 244
listviews 233
loading Web pages，加载Web页面 235
PhoneGap Build projects，PhoneGap Build项目 324～327

functions，函数
addColor() 277
buildAddButton() 277
callback, mediaCapture API 349
initDevice 339
initPhoneGap 347
matching，匹配 191
testing capabilities，测试能力 193～198
onColorListChange() 277
onStitchSizeChange() 277～278
refreshTartans 352
setColorSelectStyle() 277
styleColorListItem() 277～278
switch，切换 192
tartanFound 348, 350, 351

future-friendly manifesto，未来友好宣言 362～363
flexibility，灵活性 364
guidelines for usage，使用原则 366～370
Tartan Hunt 365

G

generate.php 280, 281

geographic trends, device support，地理趋势，设备支持 145

geolocation，地理定位 267, 270, 298～309
browser support，浏览器支持 303～308
Find Events page，查找事件页面 301, 309
forms，表单 301～308
JavaScript 300, 302
World Wide Web Consortium (W3C) Geolocation API 298

getCapability method，getCapability方法 171

getCurrentPosition method，getCurrentPosition方法 299

getDeviceForUserAgent method，getDeviceForUserAgent方法 164, 172

Global Corp 147～150, 149～150

global positioning system (GPS)，全球定位系统（GPS） 298

Google Chrome 85

Google Gears 306

Google Maps iframe code，Google Maps iframe代码
adding with JavaScript，用JavaScript增加 74～78
moving attributes to CSS，属性移动到CSS 80

scaling maps，缩放地图 82
troubleshooting widgets，部件排错 79～81
using media queries for content breaks，对内容中断使用媒体查询 83～88

GPS (global positioning system)，GPS（全球定位系统） 298

gradients, jQuery Mobile，梯度, jQuery Mobile 230～231

graphics, fluid，图像，流式 33

graphics libraries，图像库 141

groups，组
capability，能力
bearer，承载 176
device classes，设备类 181
retrieving capability values，获取能力值 171

Guda, Krishna 158

H

hardware performance, device support，硬件性能，设备支持 144

HAR (HTTP Archive) 49～50

HAR Viewer page Show Statistics link,HAR Viewer页面 Show Statistics链接 50

has_cellular_radio capability，has_cellular_radio能力 176

haz.io 383

headers，页眉
adding buttons to，增加按钮到 262～263
fixing in position，位置固定 251～252
user agent. 参见 user agents，用户代理

Head First HTML5 Programming 224

hiding JavaScript，隐藏JavaScript 53

higher_mobile devices，higher_mobile设备 186

horizontal layouts compared to vertical，水平布局与垂直布局对比 29

.htaccess files, creating cache manifests，.htaccess文件，创建缓存清单 287～288

HTML
requests for HTML pages，HTML页面请求
traditional Web，传统网站 220
Web Apps，Web应用 221
semantic markup，语义标记 57
Splendid Walrus website，非凡海象网站 15～19
weinre.html 377

HTML5 217, 218
application stores，应用商店 385
cache manifest，缓存清单
creating，创建 285～296
definition，定义 284
compared to XHTML-Mobile Profile，与XHTML-Mobile Profile对比 115～116
data-* attribute，data-*属性 231～232
data-role attribute，data-role属性 232
definition of，定义 219
forms，表单 256～257
adding buttons to headers，增加按钮到页眉 262～263
adding fields，增加域 257
color fields，颜色域 259～260
color-size field pairs，颜色-大小域对 260
lists within lists，列表中的列表 258
testing，测试 263
localStorage API 333
PhoneGap. 参见 PhoneGap
Web apps，Web应用 219

html5test.com 383

HTTP Archive (HAR) 49～50

hybrid applications，混合型应用 313, 317
browser support，浏览器支持 339
creating，创建 322～332
app.js file，app.js文件 336～352
data storage，数据存储 334～338
mediaCapture 348～353
pictures/media，图片/媒体 345～349
PhoneGap. 参见 PhoneGap
refreshing pages，刷新页面 340
tools，工具 320
uninstalling，卸载 332

I

ifconfig command，ifconfig命令 394

iframe code, YouTube videos，iframe代码，YouTube视频 35

IIS Manager 393

image.php 280

images，图像
CSS, downloading，CSS，下载 71
fluid，流式 33, 54
media queries，媒体查询 66
offline mode，离线模式 290～296
optimization，优化 67, 69
overlap，重叠 83～89

RWD (Responsive Web Design)，RWD（响应式Web设计） 10～11
scaling，缩放 68～69
Sencha.io Src 69, 71
sizes，大小 50, 52, 54
src attribute，src属性 68～69
I' m Freaking Out page，I' m Freaking Out页面 152
testing，测试 174
<img> tag，<img>标记 71
src attribute，src属性 68～69
@import syntax，@import语法 37
index.html file, Spendid Walrus website，index.html文件，非凡海象网站 15
index_mobile_.html file，index_mobile_.html文件
converting to XHTML-Mobile Profile，转换为XHTML-Mobile Profile 114～118
index.php file，index.php文件 209
index_solution.php file，index_solution.php文件 209
initDevice function，initDevice函数 339
initPhoneGap function，initPhoneGap函数 347
installing，安装
Android apps with adb，利用abd安装Android应用 411～413, 415
Android platforms，Android平台 405～406
Android SDK 320, 329, 404
Android tools，Android工具 405～406
bookmarklets 85
MAMP 392～393
Tartan Hunt 329～332
instantiating，实例化
WURFL (Wireless Universal Resource FiLe) devices，WURFL（无线通用资源文件）设备 172
WURFL (Wireless Universal Resource FiLe) objects，WURFL（无线通用资源文件）对象 163
interfaces，接口 6
Internet Explorer
conditional comments，条件注释 62
media queries，媒体查询 65
mobile media queries，移动媒体查询 62～65
testing media queries in，测试媒体查询 61～64
Internet Explorer 9 application cache，Internet Explorer 9应用缓存 284
iOS Developer program，iOS Developer程序 319
IP addresses，IP地址 394
iPhones 2
iPod Touch, testing device.php changes，iPod Touch,测试device.php改变 179
is_wireless_device capability，is_wireless_device能力 172
iWebInspector, debugging，iWebInspector,调试 381

J

JavaScript 217, 219
adding Google Maps iframe code，增加Google Maps iframe代码 74～78
adding to On Tap Now page，增加到On Tap Now页面 77～78
cache manifest，缓存清单 295
creating media queries，创建媒体查询 76
deviceready event，deviceready事件 345～346
events，事件 365
Find Events page，查找事件页面 301
geolocation，地理定位 300～308
hiding，隐藏 53
image sizes，图像大小 50
localStorage API 333
PhoneGap 319
refreshing pages，刷新页面 340
removing attributes from，从中删除属性 81
Tartanator page enhancements，花格布爱好者应用页面增强 274～279
JavaScript API, PhoneGap 345～349
JavaScript magnets, localStorage API，JavaScript磁贴, localStorage API 336～338
jQuery Core 229～230
jQuery Mobile
adding footer toolbars，增加页脚工具条 249～250
animation，动画 245
code components，代码组件 228～229
creating Tartanator page，创建花格布爱好者应用页面 228～239
creating tartans，创建花格布 240～245
custom widgets，定制部件 275
data-* attribute，data-*属性 231～232, 236
device support，设备支持 245
dialogs，对话框 306
events，事件 275
footers，页脚 237～239
linking，链接 234
list dividers，列表分割条 244

list filters，列表过滤器 244
listviews 233
loading pages，加载页面 235
PhoneGap Build projects，PhoneGap Build项目 324～327

K

Kamerman, Steve 158
keys, localStorage API，键, localStorage API 343
Koch, Peter-Paul (PPK) 189, 374

L

launching emulated devices (Android)，启动模拟设备(Android) 409～410
layouts，布局
content breaks，内容中断 83～88
fixed, disadvantages，固定，缺点 26
fluid，流式
advantages of，优点 27～28
columns，栏 37
compared to fixed layouts，与固定布局对比 24～30
context switching，上下文切换 31～32
images，图像 33
horizontal compared to vertical，水平与垂直布局对比 29
RWD (Responsive Web Design)，RWD（响应式Web设计） 10～11
Splendid Walrus website，非凡海象网站 15
linking jQuery Mobile 234
links，链接
adding，增加 75～76
making calls with，调用 176
Linux
local Web server setup，本地Web服务器建立 390～391
ports，端口 393
list dividers，列表分割条 244～249
list filters，列表过滤器 244～249
list numbering, troubleshooting，列表编号排错 132～133
list.php file，list.php文件 282
lists, adding colors，列表，增加颜色 258
listviews, jQuery Mobile 233
loading Web pages，加载Web页面
AJAX 236
jQuery Mobile 235
localStorage API 296, 333～338
browser support，浏览器支持 343
client-side feature detection，客户端特性检测 339
data storage，数据存储 343
JavaScript magnets，JavaScript磁贴 336～338
keys，键 343
toggle methods，toggle方法 341～342
updating refreshTartans function，更新refreshTartans函数 352
local Web server setup，本地Web服务器建立 387～396
accessing servers，访问服务器 394～395
Apachefriends.org 390～391
Linux 390～391
Macintoshes 392～393
PHP 390～393
ports，端口 393
Windows 390～391
XAMPP 390～391
Loch Air sponsorship 314
Tartan Hunt 参见 Tartan Hunt
logical grouping，逻辑分组 183

M

Macintoshes
installing/uninstalling Android apps，安装/卸载Android应用 413
local Web server setup，本地Web服务器建立 392～393
ports，端口 393
troubleshooting XAMPP，XAMPP排错 395
magnets (JavaScript), localStorage API，磁贴(JavaScript), localStorage API 336～338
magnets (mobile CSS), creating CSS for mobile devices，磁贴(mobile CSS)，为移动设备创建CSS 19～20
Make It! submit button，Make It!提交按钮 276
MAMP 395
filesystem access，文件系统访问 401
installing，安装 392～393
manifest attribute，manifest属性 285, 286
maps，地图
adding Google Maps iframe code with JavaScript，用JavaScript增加Google Maps iframe代码 74～78
content breaks，内容中断 83～88
hiding，隐藏 53
scaling，缩放 82

Marcotte, Ethan 10

margins, converting to fluid layout，页边距，转换为流式布局 29

markup validation，标记验证 117
  access keys，访问键 123

matchers, WURFL (Wireless Universal Resource FiLe)，匹配函数，WURFL（无线通用资源文件） 159

matches, PHP API，匹配, PHP API 212～215

matching function，匹配函数 191
  testing capabilities，测试能力 193～198

maximum-scale setting，maximum-scale设置 73

media，媒体
  fluid，流式 35, 36
  RWD (Responsive Web Design)，RWD（响应式Web设计） 10～11
  Tartan Hunt 345～349
  YouTube videos, iframe code，YouTube视频，iframe代码 35

mediaCapture API 344, 348～353
  cameras，相机 344

media features，媒体特性 12

media queries，媒体查询 65
  conditional comments，条件注释 63～65
  content breaks，内容中断 83～88
  creating in JavaScript，JavaScript中创建 76
  images，图像 66
  Internet Explorer 62～65, 65
  mobile-first Responsive Web Design，移动优先响应式Web设计 61
  testing in Internet Explorer，Internet Explorer中测试 61～64

media queries (CSS)，媒体查询 (CSS) 10～11, 13～14, 37

media types，媒体类型
  CSS 12～14
  @import syntax，@import语法 37

metadata form fields, adding to forms，元数据表单域，增加到表单 257

methods，方法
  getCapability 171
  getCurrentPosition 299
  getDeviceForUserAgent 164, 172
  localStorage API 335
  toggle 341～342
  toggleClass 341～342

Mo Better Museums 146～149, 148～150

MobileAware 157～158

mobile browsers，移动浏览器
  Ajax 281
  camera access，相机访问 316
  downloading CSS images，下载CSS图像 71
  installing bookmarklets，安装 bookmarklets 85
  linking phone numbers，链接电话号码 178
  markup validation，标记验证 117
  Opera Mini 110
    simulator，仿真器 111
  Opera Mobile 110
  plug-ins，插件 45
  redirecting，重定向 104～109
  Safari 110
  sniffing，嗅探 116
  support, client-side feature detection，支持，客户端特性检测 339
  user agents，用户代理 98～103, 102～103
  XHR 281
  XHTML-Mobile Profile 116

mobile CSS magnets, creating CSS for mobile devices，mobile CSS磁贴，为移动设备创建CSS 19～20

mobile design 参见 design

mobile devices，移动设备
  A List Apart article，A List Apart文章 374
  centers for testing mobile devices，测试移动设备的中心 375
  CMS (content management system)，CMS（内容管理系统） 95
  creating CSS for，创建CSS 14～21
    analyzing CSS，分析CSS 16～17
    analyzing site structure，分析网站结构 15～16
    converting fixed layouts to fluid layouts，固定布局转换为流式布局 24～30, 34
    identifying changes，找出改变 18
    mobile magnets，移动磁贴 19～20
    questions/answers，问题/答案 37
    steps，步骤 19～23
    updating styles.css file，更新styles.css文件 22
    viewport <meta> tag，viewport <meta>标记 22
  CSS and 9
  detecting with scripts，用脚本检测 104～107
  detection，检测 97
  frames，帧 117
  future-friendly manifesto，未来友好宣言 366
  native applications，原生应用 316

RWD (Responsive Web Design), converting fixed layouts to fluid layouts，RWD（响应式Web设计），固定布局转换为流式布局 24～30
scrolling on，滚动 119～125
sharing，共享 374
sizes，大小 6
support. 参见 devices, support
targets，目标 117
testing，测试 374～375
mobile emulators，移动模拟器 374
mobile-first Responsive Web Design，移动优先响应式Web设计 56, 57, 71
converting Responsive Web Design into，响应式Web设计转换为 58
media queries，媒体查询 61
widgets，部件 79～81
mobile frameworks，移动框架 217, 225
disadvantages，缺点 225
jQuery Mobile
adding footer toolbars，增加页脚工具条 249～250
animation，动画 245
code components，代码组件 228～229
creating Tartanator page，创建花格布爱好者应用页面 228～239
creating tartans，创建花格布 240～245
custom widgets，定制部件 275
data-* attribute，data-*属性 231～232, 236
device support，设备支持 245
dialogs，对话框 306
events，事件 275
footers，页脚 237～239
linking，链接 234
list dividers，列表分割条 244
list filters，列表过滤器 244
listviews 233
loading Web pages，加载Web页面 235
PhoneGap Build projects，PhoneGap Build项目 324～327
mobile interfaces，移动界面 6
Mobile Perf Bookmarklet 65
mobile performance tests, Mobitest，移动性能测试，Mobitest 47～53
mobile phones，移动手机 2
iPhones 2
viewing websites on，查看网站 4～7
Web browsers，Web浏览器 3
Web usage，Web使用 3
mobile simulators，移动仿真器 374
mobile sites, CSS-MP (CSS Mobile Profile 2.0)，移动网站，CSS-MP (CSS Mobile Profile 2.0) 127
mobile Web apps，移动Web应用 217, 218
definition of，定义 219
frameworks (mobile Web apps), jQuery Mobile，框架（移动Web应用），jQuery Mobile 228～239
geolocation，地理定位 267, 270
HTML5 219
mobile frameworks，移动框架 217, 225
disadvantages，缺点 225
offline mode，离线模式 267, 267～268, 270, 284～296
progressive enhancement，渐进增强 267, 270
requests for HTML pages，HTML页面请求 221
Tartans Unlimited project，Tartans Unlimited项目
description of，描述 222～225
mobile Web browsers，移动web浏览器 6
mobile websites，移动网站 104
AcedIt! Test Prep standalone website plans，AcedIt! Test Prep独立网站计划 182
mobile device detection script，移动设备检测脚本 104～107
mockups，模拟网站 108～112
single sites，单独的网站
Creature Comforts International needs，动物关怀国际组织需求 93
sending mobile users to，将移动用户指向 96
user agents，用户代理 98～103
testing，测试 109
user agents，用户代理
mobile browsers，移动浏览器 102～103
sniffing，嗅探 97, 101
strings，字符串 98～99
Mobitest 47～53
test result variations，测试结果变化 65
mockups，模拟网站
single sites，单独的网站 108～112
study aid site (AcedIt! Test Prep site)，学习辅助网站 (AcedIt! Test Prep网站)
device classes，设备类 201～209
modernizr.com 383
Mueller, Patrick 376

N

names, device classes，名字，设备类 189

native applications，原生应用 246～253, 313
device capabilities，设备能力 316
PhoneGap. 参见 PhoneGap
navbars, creating，导航条，创建 249～252
networks，网络
EDGE, 4G phones，EDGE, 4G 手机 65
WiFi, Web requests，WiFi,Web请求 45
NETWORK section，NETWORK部分 (cache manifest，缓存清单), wildcards，通配符 289, 291, 296
Nitobi 318

O

objects，对象
CustomDevice 189
window.applicationCache 294
WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件）
instantiating，实例化 163
offline mode，离线模式 267, 270
cache manifest，缓存清单 285～296
browser support，浏览器支持 296
changing resource list，修改资源列表 290～296
content-type 285
definition，定义 284
FALLBACK section，FALLBACK部分 296
.htaccess files，.htaccess文件 287
syntax，语法 285
testing，测试 295
troubleshooting，排错 292～293
URLs 285
WebKit Web Inspector 286
images，图像 290～296
Tartanator enhancements，花格布爱好者应用增强 284～296
Web pages，Web页面 290～296
onColorListChange() function，onColorListChange()函数 277
onStitchSizeChange() function，onStitchSizeChange()函数 277～278
On Tap Now page，On Tap Now页面
adding JavaScript，增加JavaScript 77～78
hiding map，隐藏地图 53
optimization issues，优化问题 44～47, 70
reordering content，内容重排序 59～60
structure，结构 58～59
testing with Mobitest，用Mobitest测试 47～53
open source licensing，开源许可
AGPL (Affero General Public License v3) 158
WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件） 158
Opera Dragonfly 65, 381
Opera Mini 110
application cache，应用缓存 284
simulator，仿真器 111
Opera Mini 4.2 120
Opera Mobile 110
optimization，优化 43, 44～47
images，图像 67, 69
On Tap Now page，On Tap Now页面 44～47, 70
ordering content，内容排序 59～60, 65
overflow: scroll property，overflow: scroll属性 131
overlapping content，重叠内容 83～89

P

<p> tag, adding links，<p>标记，增加链接 75～76
pagecreate event，pagecreate事件 275, 277～278
pageinit event，pageinit事件 275, 277～278
panic button. 参见 red panic button，紧急按钮
Passani, Luca 158
patches, WURFL (Wireless Universal Resource FiLe)，补丁，WURFL（无线通用资源文件） 213
Perfecto 375
performance，性能
4G phones，4G手机 65
hardware, device support，硬件，设备支持 144
performance tests, Mobitest，性能测试, Mobitest 47～53
phone calls, making calls with links，手机拨号，用链接拨号 176
PhoneGap 318～321, 364
Android 319
downloading APK (Android package)，下载APK（Android包） 328
creating hybrid applications，创建混合型应用 322～327
deviceready event，deviceready事件 345～346
JavaScript API 345～349
mediaCapture API 344, 348～353
page initialization，页面初始化 365

platforms，平台 320
removing references，删除引用 365
PhoneGap Build 318, 320, 321
creating Android native apps，创建Android原生应用 322～327
downloading APK (Android package)，下载APK（Android包） 328
phonegap.js 344～346
uninstalling apps，卸载应用 332
phonegap.js 344～346
PhoneGap weinre 376～381
phone numbers, linking to mobile browsers，电话号码，链接到移动浏览器 178
photos 参见 pictures
PHP
adding field groups，增加域组 258
creating cache manifest，创建缓存清单
generating files，生成文件 287～289
resource list changes，资源列表改变 290～296
local web server setup，本地web服务器建立 390～393
testing，测试 396
testing device capabilities in device class definitions，测试设备类定义中的能力 193～198
PHP API 156
capability values，能力值 173
matches，匹配 212～215
RIS matching (reduction-in-string)，RIS匹配（串内归约） 159
pictures, Tartan Hunt，图片, Tartan Hunt 345～349
pie charts，饼图 50
platforms，平台
Android 405～406
PhoneGap 320
plug-ins，插件 45
ports, local Web server setup，端口，本地Web服务器建立 393
prioritization, testing mobile devices，指定优先级，测试移动设备 375
programming, server-side，编程，服务器端 9
progressive enhancement，渐进增强 55, 57, 267, 270, 364
device support，设备支持 141
properties, overflow: scroll，属性，overflow: scroll 131
prototypes, Tartanator，原型，花格布爱好者应用 224, 253～261
proxy servers，代理服务器
Charles Proxy 65
Web requests，Web请求 45～46

Q

queries, media，查询，媒体 65
conditional comments，条件注释 63～65
content breaks，内容中断 83～88
creating in JavaScript，JavaScript中创建 76
images，图像 66
Internet Explorer 62～65
mobile-first Responsive Web Design，移动优先响应式Web设计 61
quirksmode.org 382

R

reading waterfall charts，读瀑布图 51～52
.ready event，.ready事件 275
redirecting mobile browsers，重定向移动浏览器 104～109
redirect.php file，redirect.php文件 106
red panic button，紧急按钮
AcedIt! Test Prep website，AcedIt! Test Prep网站 153
testing，测试 174～175
refreshTartans function，refreshTartans函数 352
regular expressions, mobile device detection script，正则表达式，移动设备检测脚本 107
remote debugging、远程调试 376～381
remote devices, testing，远程设备，测试 375
removing attributes from JavaScript，从JavaScript中删除属性 81
reordering，重排序，参见 ordering content
requests for web pages，Web页面请求
asynchronous requests，异步请求 221
traditional website behavior，传统网站行为 220
resource path, WURFL PHP API，资源路径, WURFL PHP API 402
resources，资源
cache manifest, listing，缓存清单，列出 289
changing in cache manifest，缓存清单中修改 290～296
downloading waterfall charts，下载瀑布图 48, 49～50
Responsive Design and Server-Side components (RESS)，响应

式设计和服务器端组件（RESS） 386
Responsive Web Design (RWD)，响应式Web设计（RWD） 1, 10～11, 41
context switching，上下文切换 31～32
converting fixed layouts to fluid layouts，固定布局转换为流式布局 24～30
CSS media queries，CSS媒体查询 10～11, 13～14
final CSS file analysis，最终的CSS文件分析 38～40
fluid layouts，流式布局 24～30
fluid media，流式媒体 35, 36
images，图像 10～11
layouts，布局 10～11
media，媒体 10～11
RESS (Responsive Design and Server-Side components)，RESS（响应式设计和服务器端组件） 386
Rieger, Bryan 386
Rieger, Stephanie 386
RIS (reduction-in-string) matching, PHP API，RIS（串中）匹配，PHP API 159
RWD (Responsive Web Design)，RWD（响应式Web设计） 1, 10～11, 41
context switching，上下文切换 31～32
converting fixed layouts to fluid layouts，固定布局转换为流式布局 24～30
CSS media queries，CSS媒体查询 10～11, 13～14
final CSS file analysis，最终的CSS文件分析 38～40
fluid layouts，流式布局 24～30
fluid media，流式媒体 35, 36
images，图像 10～11
layouts，布局 10～11
media，媒体 10～11

S

Safari 110
testing cache manifests，测试缓存清单 286
testing strings，测试字符串 189
troubleshooting cache manifest，缓存清单排错 291
scaling，缩放
disabling，禁用 73
Google Map maps，oogle Map地图 82
images，图像 68～69
viewport <meta> tag，viewport <meta>标记 72
ScientiaMobile Explorer 155, 158
APIs 158
creating explore page，创建资源管理页面 160～166, 169
testing explore page，测试资源管理页面 169
scripts, mobile device detection，脚本，移动设备检测 104～107
<script> tag，<script>标记 377
scrolling access key attributes，滚动访问键属性 119, 123～125
SDKs (software development kits), Android，SDKs (软件开发包), Android 319, 403
adb 411～413, 415
adding emulated devices，增加模拟设备 407～408
downloading，下载 404
installing，安装 320, 329, 404
launching emulated devices，启动模拟设备 409～410
platforms/tools，平台/工具 405～406
semantic markup，语义标记 57, 364
Sencha.io Src 69, 71, 364
servers，服务器
proxy，代理
Charles Proxy 65
Web requests，Web请求 45～46
weinre 377～379
server-side device detection，服务器端设备检测 150
server-side programming，服务器端编程 9
services, device support，服务，设备支持 144
setColorSelectStyle function，setColorSelectStyle函数 277
SGML (Standard Generalized Markup Language)，SGML（标准通用标记语言） 116
shared structural CSS，共享结构CSS 21
sharing devices，共享设备 374
Show Statistics link (HAR Viewer page)，Show Statistics链接 (HAR Viewer页面) 50
simpler_mobile devices，simpler_mobile设备 186
simulators，仿真器 374
single sites，单独的网站 93, 104
AcedIt! Test Prep standalone website plans，AcedIt! Test Prep独立网站计划 182
CSS-MP (CSS Mobile Profile 2.0) 127
mobile device detection script，移动设备检测脚本 104～107
mockups，模拟网站 108～112
scrolling on mobile devices，移动设备上滚动 119～125
sending mobile users to，将移动用户指向 96
testing，测试 109
user agents，用户代理 98～103
mobile browsers，移动浏览器 102～103

sniffing，嗅探 97, 101
strings，字符串 98～100
size field，大小域 260
sizes，大小
files，文件 52
HAR Viewer page pie charts，HAR Viewer页面饼图 50～51
fonts，字体 35
em-based 37
images，图像 50, 52, 54
src attribute，src属性 68～69
jQuery Mobile 230～231
mobile devices，移动设备 6
Skyfire, user agent string，Skyfire,用户代理串 103
smartphones，智能手机
Android 3
compared to feature phones，与功能手机比较 113, 120
sniffing，嗅探
browsers，浏览器 116
user agent，用户代理 97, 101, 135
software development kit (SDK) 参见 SDKs
speed，速度
connection, device support，连接，设备支持 144
troubleshooting，排错 44～47
Splendid Walrus website，非凡海象网站
index.html file，index.html文件 15
mobile phone view，移动手机视图 4～9
On Tap Now page，On Tap Now页面
adding JavaScript，增加JavaScript 77～78
hiding map，隐藏地图 53
optimization issues，优化问题 44～47
reordering content，内容重排序 59～60
structure，结构 58～59
testing performance，测试性能 47～53
optimization，优化 44～47
structure，结构 15
styles.css file，styles.css文件 16
spoofing browser user agents，诱骗浏览器用户代理 103
src attribute，src属性
CSS 68
<img> tag，<img>标记 68～69
stakeholders, device support，投资人，设备支持 144
Standard Generalized Markup Language (SGML)，标准通用标记语言（SGML） 116
standards，标准
future-friendly manifesto，未来友好宣言 366
localStorage API 333～338
stores (application)，商店（应用） 385
storing data, localStorage API，存储数据, localStorage API 334～338
strings，字符串
localStorage API 343
user agents，用户代理 98～100
mobile device detection script，移动设备检测脚本 107
Skyfire 103
xhtml_make_phone_call_string 176
structure, On Tap Now page，结构，On Tap Now页面 58～59
study aid site mockup (AcedIt! Test Prep site)，学习辅助模拟网站(AcedIt! Test Prep网站)
device classes，设备类 201～209
styleColorListItem function，styleColorListItem函数 277～278
styles.css
CACHE section，CACHE部分 (cache manifest，缓存清单) 290
Splendid Walrus Web，非凡海象网站 16
testing，测试 133～134
updating，更新 22
super mobile Web apps，超级移动Web应用 267参见 Web应用
Tartanator enhancements，花格布爱好者应用增强 270～280
backend enhancements，后端增强 280～283
forms，表单 274～277
geolocation，地理定位 298～309
offline mode，离线模式 284～296
support，支持
browsers，浏览器
caniuse.com 382
client-side feature detection，客户端特性检测 339
gelocation，地理定位 303～308
localStorage 343
testing，测试 383
devices，设备 137, 139
analytics，分析 145
audience considerations，用户考虑 142～145
browser features，浏览器特性 144
browsers，浏览器 140
connection speed，连接速度 144
customer considerations，客户考虑 150
demographics，人口统计 145
determining criteria/priorities，确定准则/优先级 138
geographic trends，地理趋势 145
hardware performance，性能 144
jQuery Mobile 245

progressive enhancement，渐进增强 141
questions to ask，要问的问题 142～143
services，服务 144
stakeholders，投资人 144
unsupported devices，不支持的设备 140～141
swatch c, footers，swatch c, 页脚 239
switch statement，switch语句 192
syntax, cache manifest，语法，缓存清单 285

T

tablets，平板电脑 185
tags，标记
<div>, adding links，<div>,增加链接 75～76
<img> 71
src attribute，src属性 68～69
<meta> 72～73, 89
<p>, adding links，<p>,增加链接 75～76
<script> 377
 240～243
viewport <meta> 22, 72～73, 89
taps.css, floating content，taps.css, 浮动内容 60
targets, mobile devices，目标，移动设备 117
Tartanator，花格布爱好者应用 268～269
building with jQuery Mobile，用jQuery Mobile构建 228
data-* attribute，data-*属性 231～232
creating tartans，创建花格布 240～245
description of project，项目描述 222～225
events，事件 271～272
Events page，事件页面 271～272
forms，表单 253～261
Phase 2 (super mobile Web app)，第2阶段（超级Web移动应用） 270～280
backend enhancement，后端增强 280～283
form enhancement，表单增强 274～276
geolocation，地理定位 298～309
offline mode，离线模式 284～296
prototype，原型 224, 253～261
welcome message，欢迎消息 271～272
tartanFound function，tartanFound函数 348, 350, 351
Tartan Hunt 315, 365
creating，创建 322～332
app.js file，app.js文件 336～352
mediaCapture 348～353
pictures/media，图片/媒体 345～349
refreshing pages，刷新页面 340
showing found tartans，显示找到的花格布 340
storing found tartans，存储找到的花格布 334～338
data storage，数据存储 334～338
installing，安装 329～332
showing found tartans，显示找到的花格布 340
storing tartans，存储花格布 334～338
tartans.html file，tartans.html文件 240
tartans.php file，tartans.php文件 282
Tartans Unlimited
About Us page，关于我们页面 271
description of，描述 222～225
Loch Air sponsorship 314
Tartan Hunt 参见 Tartan Hunt
Tera-WURFL, ScientiaMobile Explorer 155
testing，测试
access keys，访问键 125
Android apps，Android应用 414
Android string，Android串 189
app.js file，app.js文件 353～355
browser support，浏览器支持 383
cache manifest，缓存清单 286, 295
centers for testing mobile devices，测试移动设备的中心 375
CSS 23
device capabilities in device class definitions，设备类定义中的设备能力 193～198
device classes，设备类 200, 209
devices，设备 139
explore page，资源管理页面 169
HTML5 forms，HTML5表单 263
I’m Freaking Out page，I’m Freaking Out页面 174
media queries in Internet Explorer，Internet Explorer中媒体查询 61～64
mobile devices，移动设备 374～375
performance, Mobitest，性能, Mobitest 47～53
PHP 396
prioritization在，指定优先级 375
remote devices，远程设备 375
Safari string，Safari串 189
single sites，单独的网站 109
styles.css 133～134
theme CSS 229～230
toggleClass method，toggleClass方法 341～342
toggle method，toggle方法 341～342
toolbars (footer), adding to headers/footers with jQuery Mobile，工具条（页脚），用jQuery Mobile增加到页眉/页脚

249～252
tools，工具
Android 405～406
hybrid applications，混合型应用 320
troubleshooting，排错
access keys，访问键 122, 123～125, 125
Apache Friends Support Forum 394
appCache 289, 291
cache manifest，缓存清单 291, 292～293
device classes，设备类 198
list numbering，列表编号 132～133
mediaCapture API 349
remote debugging，远程调试 376～381
speed issues，速度问题 44～47
widgets，部件 79～81
Twitter API 95

U

UAProf (User Agent Profile) 103
 tag，<ul>标记 240～243
uninstalling，卸载
Android apps with adb，用abd卸载Android应用 411～413, 415
hybrid apps，混合型应用 332
updates, WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件） 158
URLs, cache manifest，URL，缓存清单 285
User Agent Profile (UAProf) 103
user agents，用户代理 98～103
browsers, spoofing，浏览器，诱骗 103
facebookexternalhit 213
mobile browsers，移动浏览器 102～103
sniffing，嗅探 97, 101, 135
strings，字符串 97, 98～100
mobile device detection script，移动设备检测脚本 107
Skyfire 103

V

validating code，验证代码 374
validation，验证 117
access keys，访问键 123
vertical layouts compared to horizontal垂直布局与水平布局对比 29
videos, YouTube iframe code，视频，YouTube iframe代码 35
viewport <meta> tag，viewport <meta>标记 22, 72～73, 89
viewports，端口 89
viewport <meta> tag，viewport <meta>标记 72, 89
virtual devices，虚拟设备
installing in Android SDK，Android SDK中安装 407～408
launching (Android)，启动(Android) 409～410

W

W3C device APIs 384
W3C Geolocation API，W3C地理定位API 298
browsers，浏览器 299～300
W3C mediaCapture API 344
WAC (Wholesale Applications Community) 384
waterfall charts，瀑布图 48
analyzing image sizes，分析图像大小 54
HAR (HTTP Archive) 49～50
reading，读 51～52
Web applications, creating hybrid applications，Web应用，创建混合型应用 322～327
Web apps，Web应用 217, 218参见 super mobile Web apps
Android 403
installing，安装 415
installing with adb，用adb安装 411～413
testing，测试 414
uninstalling，卸载 415
uninstalling with adb，用adb卸载 411～413
backend enhancements，后端增强 280～283
definition of，定义 219
form enhancements，表单增强 274～277
frameworks (mobile Web apps), jQuery Mobile，框架（移动Web应用），jQuery Mobile 228～239
geolocation，地理定位 267, 270
HTML5 219
hybrid applications，混合型应用 313, 317
creating，创建 322～332
tools，工具 320
mobile frameworks，移动框架 217, 225
disadvantages，缺点 225
native applications，原生应用 313
device capabilities，设备能力 316
offline mode，离线模式 267, 270, 284～296
progressive enhancement，渐进增强 267, 270
requests for HTML pages，HTML页面请求 221
Tartans Unlimited project, description of，Tartans Unlimited

项目，描述 222～225

Web browsers，Web浏览器
- application cache，应用缓存 284
- Chrome
  - testing cache manifests，测试缓存清单 286
  - troubleshooting cache manifest，缓存清单排错 291
- device support，设备支持 140, 144
- Internet Explorer
  - conditional comments，条件注释 62～63
  - mobile media queries，移动媒体查询 62～65
  - queries media，查询媒体 65
  - testing media queries in，测试媒体查询 61～64
- mobile phones，移动手机 3
- mobile web browsers，移动web浏览器 6
- Safari
  - testing cache manifests，测试缓存清单 286
  - troubleshooting cache manifest，缓存清单排错 291
- support, cache manifest，支持，缓存清单 296

Web, cross-platform technology，Web，跨平台技术 5

Web design，Web设计 357
- determining which design to use，确定使用哪个设计 358～361
- fluid layouts compared to fixed layouts，流式布局与固定布局比较 24～30
- future-friendly manifesto，未来友好宣言 362～363
  - flexibility，灵活性 364
  - guidelines for usage，使用原则 366～370
  - Tartan Hunt 365
- mobile-first responsive，移动优先响应 56～57, 71
  - converting Responsive Web Design into，响应式Web设计转换为 58
  - media queries，媒体查询 61
  - widgets，部件 79～81
- progressive enhancement，渐进增强 55, 57, 141
- RWD (Responsive Web Design)，RWD（响应式Web设计） 10～11
  - CSS media queries，CSS媒体查询 10～11, 13～14
  - images，图像 10～11
  - layouts，布局 10～11
  - media，媒体 10～11

Web Developer Toolkit，Web开发包 85

WebGL 141

Web Inspector 65, 286

WebKit
- browsers，浏览器 189
- debugging，调试 381
- Web Inspector, creating cache manifests，Web Inspector，创建缓存清单 286

Web pages，Web页面
- offline mode，离线模式 290～296
- requests for HTML，HTML请求 220

Web requests, WiFi networks，Web请求，WiFi网络 45～46

Web server setup，Web服务器建立 387～396
- accessing servers，访问服务器 394～395
- Apachefriends.org 390～391
- Linux 390～391
- Macintoshes 392～393
- PHP 390～393
- ports，端口 393
- Windows 390～391
- XAMPP 390～391

Web，网站
- AcedIt! Test Prep 152
  - red panic button，紧急按钮 153
- Apachefriends.org 390～391
- browserscope.org 383
- caniuse.com 382
- debug.phonegap.com 377, 381
- Device Anywhere 375
- html5test.com 383
- mobile, user agent，移动，用户代理 98～100
- modernizr.com 383
- Perfecto 375
- quirksmode.org 382
- requests for HTML pages，HTML页面请求 220
- RWD (Responsive Web Design)，RWD（响应式Web设计） 1, 10～11
- sharing information，共享信息 93
- single sites，单独的网站 93, 104
  - AcedIt! Test Prep standalone website plans，AcedIt! Test Prep独立网站计划 182
  - mobile device detection script，移动设备检测脚本 104～107
  - mockups，模拟网站 108
  - scrolling on mobile devices，移动设备上滚动 119～125
  - sending mobile users to，将移动用户指向 96
  - user agents，用户代理 98～103
  - user-agent sniffing，user-agent嗅探 97, 101
- Splendid Walrus，非凡海象网站
  - index.html file，index.html文件 15
  - mobile phone view，移动手机视图 4～9
  - optimization，优化 44～47
  - structure，结构 15～19
  - styles.css file，styles.css文件 16
  - testing performance of On Tap Now page，On Tap Now页面的测试性能 47～53

Tartans Unlimited 222～225

Web standards, localStorage API，Web标准，localStorage API 333～338

Web technologies，Web技术 6

weinre 65, 376～381

weinre servers，weinre服务器 377～379

welcome message, Tartanator，欢迎消息，花格布爱好者应用 271～272

Wholesale Applications Community (WAC) 384

widgets，部件

jQuery Mobile 275

Responsive Web Design，响应式Web设计 79～80

troubleshooting，排错 79～81

WiFi networks, web requests，WiFi网络，Web请求 45～46

wildcards，通配符

CACHE section，CACHE部分 (cache manifest，缓存清单) 296

NETWORK section，NETWORK部分 (cache manifest，缓存清单) 289, 291

window.applicationCache object，window.applicationCache对象 294

windows，窗口

fixed layouts，固定布局 26

fluid layouts，流式布局 27

Windows

Apache Friends Support Forum 394

installing/uninstalling Android apps，安装/卸载Android应用 413

local Web server setup，本地Web服务器建立 390～391

ports，端口 393

WIP (Wireless Industry Partnership) 385

wireless CSS，无线CSS 135

Wireless Industry Partnership (WIP) 385

Wireless Markup Language (WML) 117

Wireless Universal Resource FiLe 参见 WURFL

WML (Wireless Markup Language) 117

World Wide Web Consortium (W3C) Geolocation API. 参见 W3C Geolocation API，W3C地理定位API

Wroblewski, Luke 386

WURFL (Wireless Universal Resource FiLe)，WURFL（无线通用资源文件） 155～159

APIs 159, 213

capabilities，能力 156～157, 170～171

creating device classes，创建设备类 187～189, 190～199

creating links，创建链接 176

data，数据 159

instantiating devices，实例化设备 172

matchers，匹配函数 159

objects, instantiating，对象，实例化 163

open source licensing，开源许可 158

patches，补丁 213

PHP API 156

updates，更新 158

WURFL PHP API 397

code，代码 402

downloading，下载 398

filesystem access，文件系统访问 401

resource path，资源路径 402

X

XAMPP

Apache Friends Support Forum 394

downloading，下载 390

local Web server setup，本地Web服务器建立 390～391

XHR 281

browser support，浏览器支持 281

XHTML-Basic 119, 135

xhtml_make_phone_call_string 176

XHTML-Mobile Profile 114～118

advantages of，优点 117

compared to HTML5，与HTML5比较 115～116

mobile browsers，移动浏览器 116

XML

data files，数据文件 397, 399

data representation storage，数据表示存储 282

Y

YouTube, iframe code，YouTube, iframe代码 35

Z

zooming，放大 89

disabling，禁用 73

viewport <meta> tag，viewport <meta>标记 72～73

郑重声明

最值得程序员珍藏的200部技术经典系列仅供程序员内部学习交流使用，未经允许不得用于任何商业用途。感谢您的配合！